HA100

SAP HANA Introduction

INSTRUCTOR HANDBOOK
INSTRUCTOR-LED TRAINING

Course Version: 15
Course Duration: 2 Day(s)
Material Number: 50147321

SAP Copyrights and Trademarks

Typographic Conventions

American English is the standard used in this handbook.

The following typographic conventions are also used.

This information is displayed in the instructor's presentation	
Demonstration	
Procedure	
Warning or Caution	
Hint	
Related or Additional Information	
Facilitated Discussion	
User interface control	*Example text*
Window title	*Example text*

Contents

Course Overview

TARGET AUDIENCE

This course is intended for the following audiences:

- Application Consultant
- Project Stakeholder

UNIT 1 Describing SAP HANA

Lesson 1

Lesson 2

UNIT OBJECTIVES

- Understand the need for a modern digital platform
- Describe how SAP HANA powers a digital platform

Understanding the Need for a Modern Digital Platform

3

LESSON OVERVIEW

This lesson will help you to appreciate the demands being placed on organizations caused by an increasingly digital world. The lesson will also help you to understand why business software needs to keep up with the fast changing technology.

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Understand the need for a modern digital platform

Digital World

In a changing world we are more connected than ever before, with more valuable data being generated every day. More services are moving online, and this trend is set to grow exponentially. Forward thinking organizations are already taking steps to adapt to the new digital world and grow their businesses.

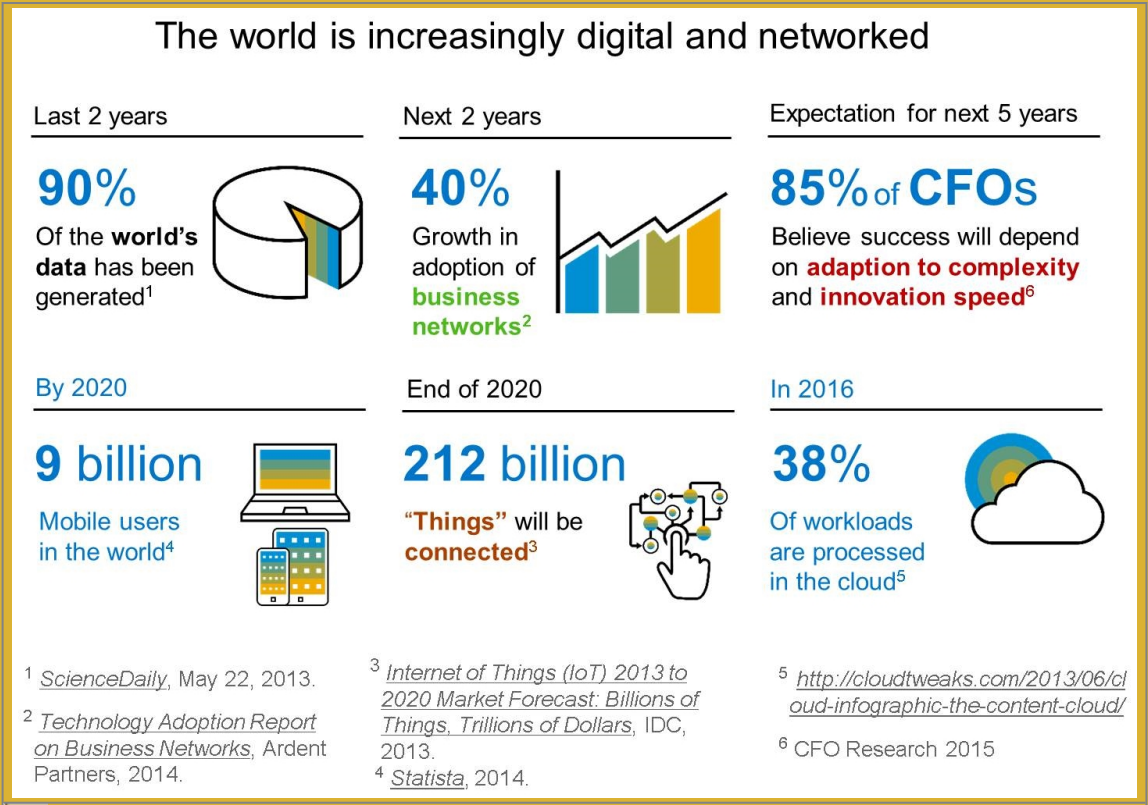


Figure 1: Increasingly Digital and Networked World

The exponential growth of mobile devices, social media, cloud technologies, and the data they generate has transformed the way we live and work. 61% of companies report that the majority of their people use smart devices for everything from e-mail to project management to content creation.

All of this development creates unprecedented opportunities for all organizations to grow their businesses. They can do this by exploiting the connectivity of consumers and business partners, tapping into the depth and variety of new types of data, acquiring this data in real time for real-time decision making, and developing innovative new applications quickly.

Increasingly Digital World

Consumerization is driving expectations of what business IT should offer for its users. As business users become familiar with smart consumer applications in their private lives, they also demand real-time, innovative applications. These applications are needed to enable deep insight and provide proactive decision support in their jobs.

However, current business systems cannot cope with the increasing demands of the digital world. We cannot just keep adding more complexity to existing IT landscapes in the hope that we can keep pace with trends. What is needed is a fresh start. It is time to start with a blank canvas and rebuild the business systems from the bottom up. This fresh start must use only the latest technologies aligned to the modern digital world. This is exactly what SAP have done by developing SAP HANA, a brand new platform built for the digital world.

Stifled Growth Due to IT Complexity

Typical IT landscapes have developed over time into multiple complex arrangements of purchased, acquired with developed applications, or powered by multiple platforms. These platforms can be based on incompatible hardware from different suppliers. This can mean different operating systems and different databases, and even different development languages. To pull together these different applications together, organizations have created interfaces between systems that need constant maintenance as individual systems evolve.

As organizations move towards cloud based applications, these need to be added to an organization's existing landscape of on-premise and other cloud based applications from different vendors. This adds increasing complexity to a landscape where multiple technology platforms are used.

Current Landscapes: Problems

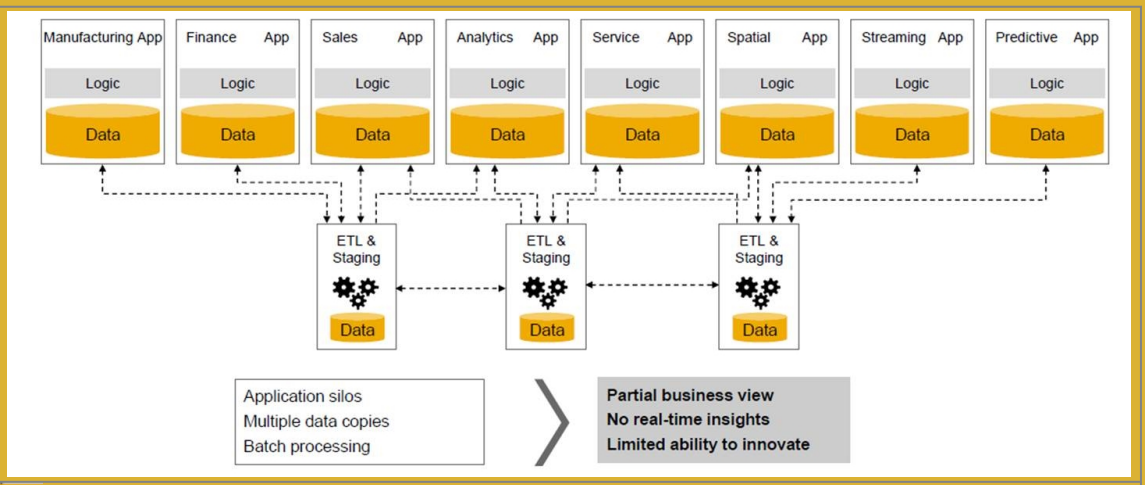


Figure 2: Current Landscapes: Problems

The IT department has been responsible for the integration of these systems. Moving, harmonizing, and cleaning data results in multiple copies of that data. We have placed huge demands on system resources during batch processing. This means that users have to wait for long running processes such as financial close, consolidations, and Materials Requirement Planning (MRP).

Complex landscapes create fragmented business views of data. To obtain a holistic view, users are required to wait until consolidation is complete. Developing new applications in a complex landscape is also difficult. It takes time and is expensive to build and maintain.

There is too much IT complexity in most organizations. Complex landscapes are costly to maintain with multiple skills needed.

Complexity is stifling growth and suppresses agility and innovation, which is critical to survival in the digital world.

One Platform for all Applications

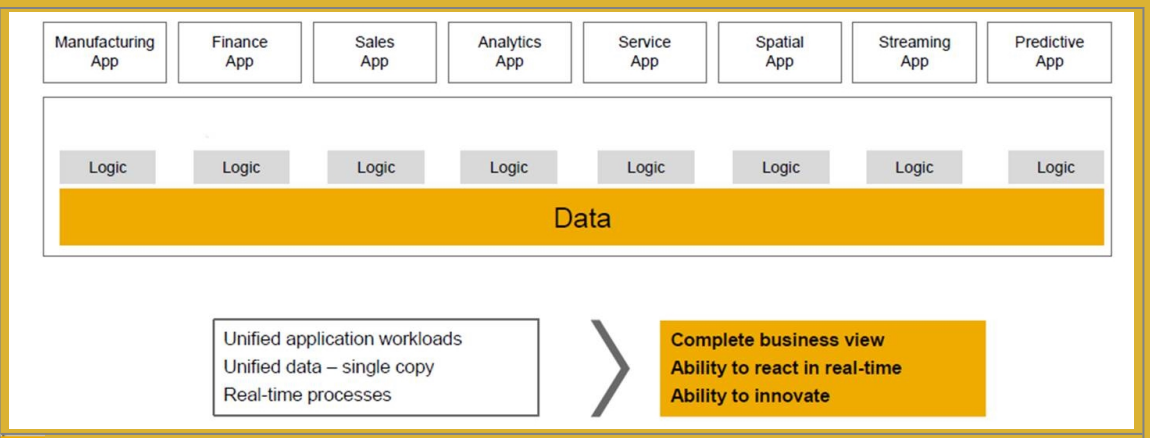


Figure 3: One Platform for all Applications

The answer is to have all applications powered by one high performance platform. This means a common architecture with only one data management solution where all data is available to all applications in real time. Then there is no more unnecessary data movement or management of multiple data stores and no more complex platform-to-platform architecture with high maintenance interfaces.

SAP HANA is the platform that simplifies an IT landscape and removes complexity that has developed over many years.

Note:
Later in the course we discuss multi-tenancy and containerization and how this is used to isolate multiple application running on one SAP HANA platform.

Consolidation of OLTP and OLAP to Create HTAP

A key objective of SAP HANA is to remove all redundancies. This means that only one copy of data is needed for any type of access.

Traditionally, systems were either optimized for transactions (OnLine Transaction Processing = OLTP) or analysis (OnLine Analytical Processing = OLAP). Transactions were managed in systems where both the hardware architecture, database design and the data models were built around fast read/write processing at the record level. Analysis systems took on a different design approach. The hardware, database, and data models were built around batch

loading, aggregated storage, and a focus on read-intensive queries and caching. That is why, historically, OLTP and OLAP were separated and linked through interfaces.

Now SAP HANA is able to bring transactional and analysis requirements into one platform. The acronym for this type of consolidated system is Hybrid Transaction / Analytical Processing (HTAP).

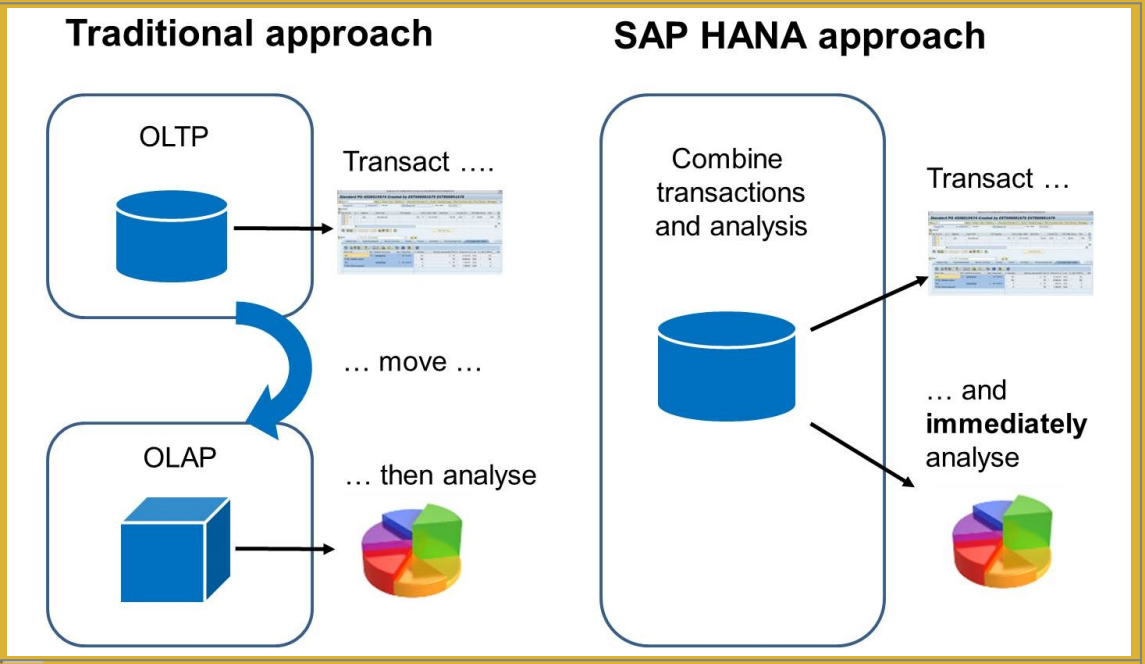


Figure 4: Consolidating OLTP and OLAP

SAP HANA is able to combine transactional and analysis processing in one platform. The database, hardware, and data model of SAP HANA are built to handle combined transactional and analysis processing. No movement of data is necessary and you always work from the same, single copy of the data for any requirement. This is true for both transactional and analytical requirements. This means that you have live data available to all applications. This reduces the complexity by removing the need to move data using separate software. It also means that applications can be built that combine transactions and analytics in one. For example: an employee vacation booking system that calculates and displays forecasted peak workload times so the employee can be sure to avoid booking vacation at a time that might impact the business. In the past two separate applications would need to have been built.

Exploiting Technological Advancement

Some of the questions that may arise include “How can one platform handle all applications?” and “Why did we not do this earlier?”

SAP HANA has been developed from scratch to take advantage of the recent trends and advances in hardware technology. SAP HANA was not built by taking existing software and building on top of it. This redevelopment was undertaken to ensure that it is able to handle such an ambitious challenge and provide a next generation platform that aligns with new approaches.

Historically, the high cost of memory meant that only small amounts were available to use. This caused a serious bottleneck in the flow of data from the disk all the way to CPU. It did not matter how fast the CPU was if the data could not reach it quickly.

We now have access to huge amounts of cheap memory. SAP HANA can run on hardware with many terabytes of memory. With so much memory available, you can store the entire

database of even the largest organizations, completely in memory. This gives you instant access to all data, and eliminates wait times caused by data traveling from disk to memory. We can lose the mechanical spinning disk and the latency it brings, and rely on memory to provide all data instantly to the CPU. Memory is no longer the bottleneck it once was. To address large amounts of memory, we also use 64-bit operating systems. Traditional 32-bit operating systems cannot address the large amounts of memory now available.

In addition to huge memory, processors continue to improve at a phenomenal rate. You now have high-speed, multi-core CPUs that can take on complex tasks and break them up so then can be processed in parallel to provide incredibly response times. This means that response times for even the most complex analytical tasks, such as predictive analysis, can be carried out in real time. Multiple CPUs that support multiple cores now gives you access to huge processing power.

Recent Advances in Technology

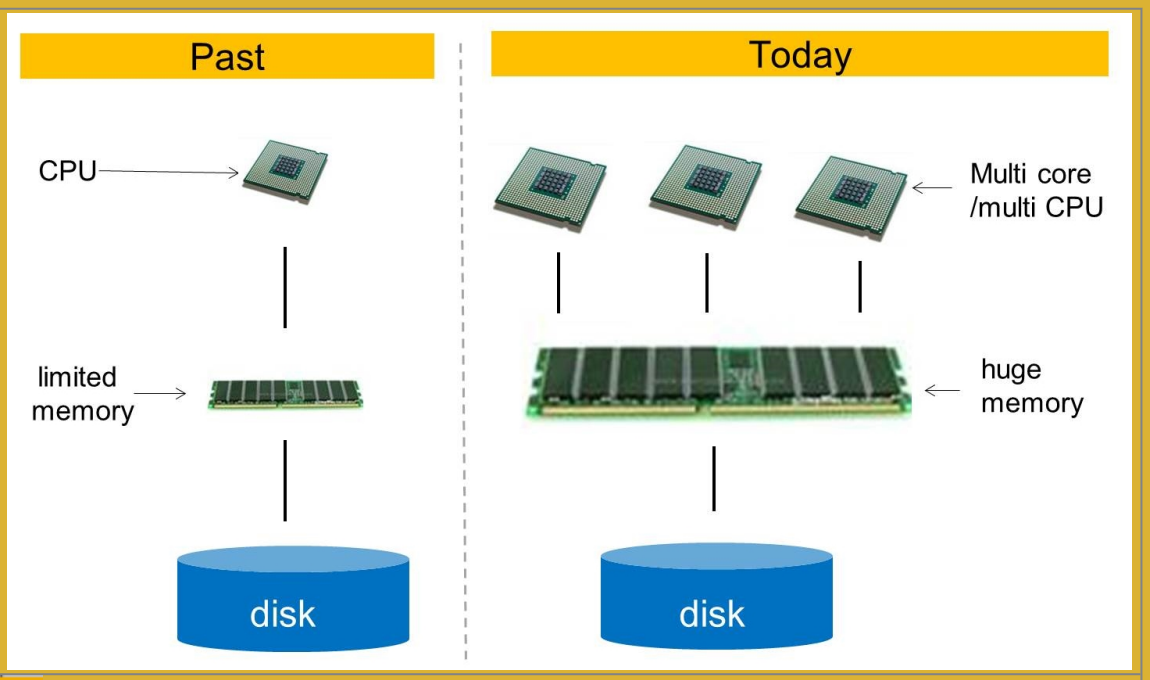


Figure 5: Recent Advances in Technology

Recent advances in the design of on-board cache mean that data can now pass between memory and CPU cores rapidly. In the past, even with large amounts of memory, as the CPUs were demanding more data the cache would struggle to keep up as the journey from memory to CPU was not optimal. We now have sophisticated on-board, multi-level CPU cache that keeps the most useful data closest to the CPU, and avoids reading from memory unless absolutely necessary. If you think of data in memory as ‘hot’ then data in cache would be regarded as ‘red hot’.

With modern blade-server architecture, you can now add more RAM and more CPUs into your landscape easily. This adds more processing power or memory, allowing you to scale up to handle huge workloads and data volumes.

It would have been possible for SAP to have kept the same business application software that was written 20 years ago, along with the traditional databases that supported them, and installed all this on the new powerful hardware. This would provide some gains, but traditional databases and applications were designed around old, restricted hardware architecture available at that time. This meant they would not be able to fully exploit the power of the new hardware, with all the new developments previously mentioned. It would be like putting a

racing driver with limited skills, into a Formula 1 race car where the potential of the car was never realized.

Put simply, the business software needed to catch up with advances in hardware technology. Thus, a complete rewrite of the platform (SAP HANA), as well as the applications that run on the platform, was required.

SAP built SAP HANA to fully exploit the latest hardware. SAP collaborated with leading hardware partners who shared the designs of their new CPU and cache architectures. This enabled SAP to develop SAP HANA in such a way that it could extract every last drop of power from the hardware.

LESSON SUMMARY

You should now be able to:

- Understand the need for a modern digital platform

9 Describing how SAP HANA Powers a Digital Platform

LESSON OVERVIEW
You want to develop a high level understanding of SAP HANA so that you can assess its potential for transforming existing application and developing new ones.

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Describe how SAP HANA powers a digital platform

Total Database Transfer to Memory

In the past, databases were stored completely on disk. Only the data requested by the applications would be moved to memory, where it then passed to the CPU for processing. Data in memory would be constantly displaced with new data requests, so a lot of swapping was normal. With SAP HANA, you can now store the complete database in memory. This means that disk movement is not needed and swapping can be eliminated.

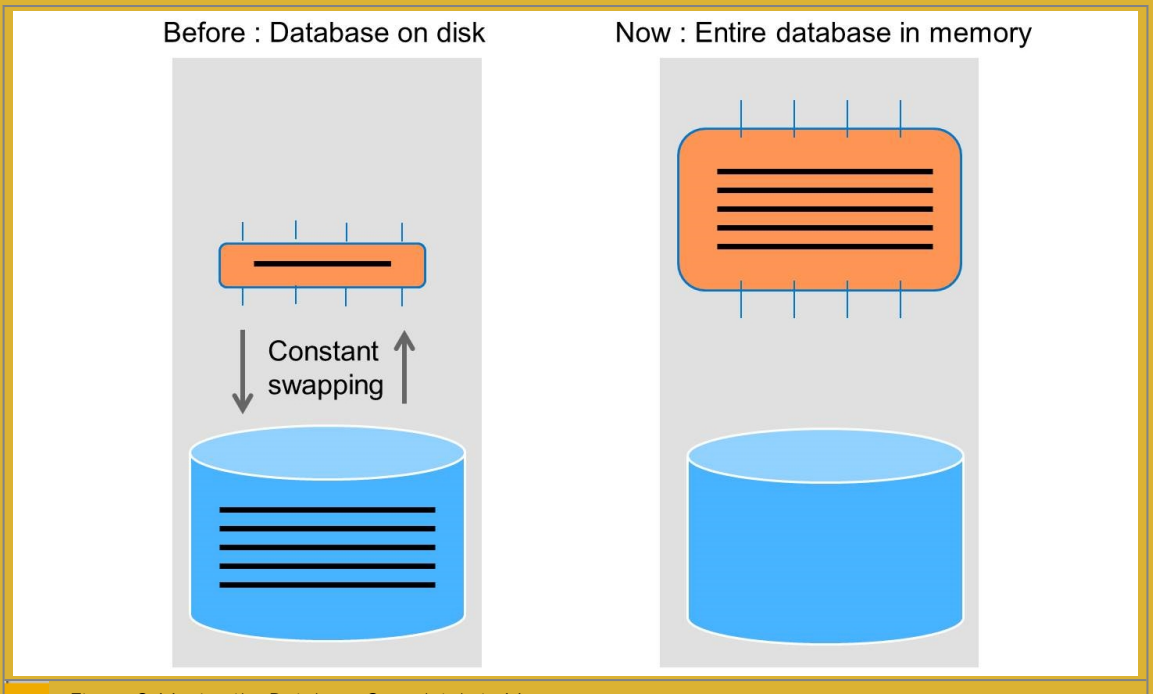


Figure 6: Moving the Database Completely to Memory

You can transfer the entire database to memory due to the following important advances:

- Huge amounts of memory are now available. We have moved from gigabytes of memory to terabytes of memory.

- SAP HANA automatically compresses data. This compression reduces the data footprint of the largest databases down to a fraction of their original size.

Note:
Since 2006, Business Warehouse Accelerator (BWA) also moved large amounts of BW disk-based data to memory to improve read performance. However, BWA could never move the entire BW data to memory, only selected InfoCubes. So, you had to make a choice as to which BW data you wanted to accelerate. SAP HANA accelerates all data because all data can fit in memory.

SAP HANA Disk Store

However, this does not mean that disk is no longer needed. SAP HANA includes disk store.

You need disk for the following reasons:

- Data in memory is referred to as hot, which means it is highly used and needs to be closest to the CPU for optimum read performance. Infrequently used data can be classified as warm, which means it is stored on disk. SAP HANA always attempts to store all data in memory. However, most organizations would not want all data in memory as they regard only a part of it to be hot. The warm data can wait on disk until it is needed, at which time it is called into memory. Of course, this means a slight delay in getting data to the CPU when compared to the hot data, but for data that is warm, this is usually acceptable. This means that you can deliberately size memory optimally to fit only the hot data and not worry about trying to fit the entire data of the organization in memory.
- Disk is used as a safe backup of memory, in case of power outage. SAP HANA regularly saves the entire contents of memory to disk so that when power is restored, memory can quickly be restored from disk.

Both of these requirements are covered in more detail later in this course. For now, it is important not to ignore disk. While it is no longer needed for reading data where high performance is required, it does still have its place in SAP HANA.

Simplified Data Models and Applications

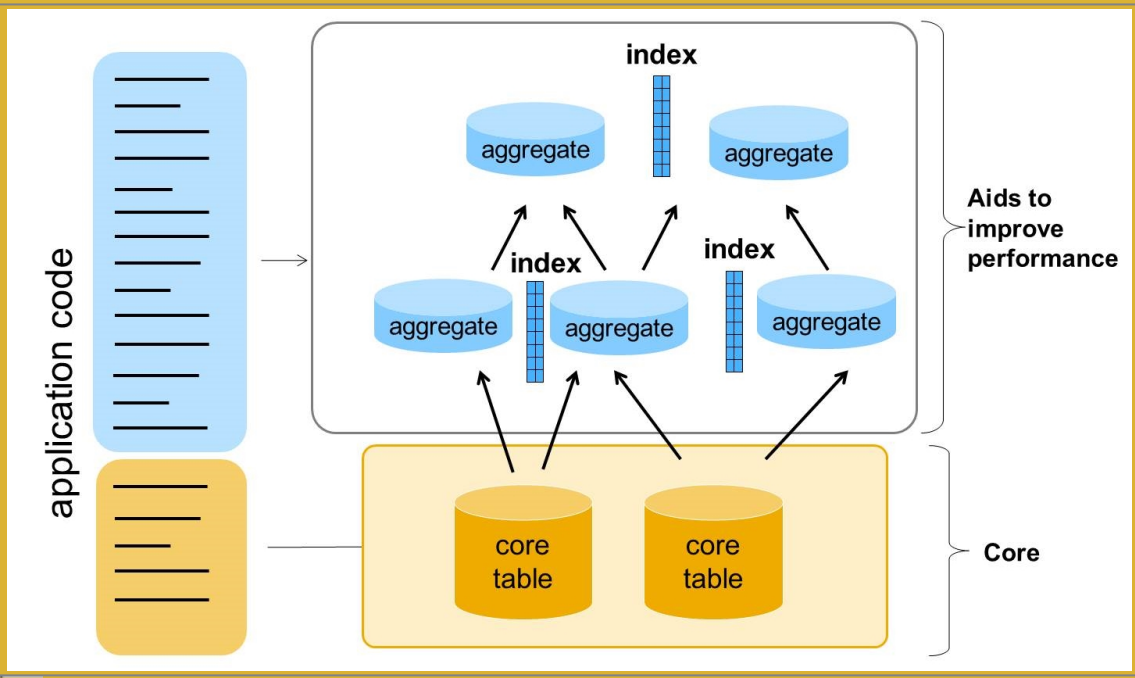


Figure 7: Traditional Approach to Data Model and Application Code

Traditional applications were built on a hierarchical data model. Detailed data was summarized into higher level layers of aggregates to help system performance. On top of aggregates, we built more aggregates and special versions of the database tables to support special applications. As well as storing the extra copies of data, we also had to build application code to maintain extra tables and keep it up to date. A backup to these extra tables was also required, so even the IT operations were impacted.

In addition to aggregates, we have another inefficiency that we need to remove. Database indexes improve access speed as they are based on common access paths to data. However, they need to be constantly dropped and rebuilt each time the tables are updated. So again, more code is needed to manage this process.

The traditional data model is complex and this causes the application code to be complex. 70% of application code is built specifically for performance of an application and adds no value to the core business function.

A complex data model and complex code, means that integration with other applications, and also enhancements, are difficult. This means that they are simply not agile enough for the modern fast moving business environment.

Eliminate Complexity of Applications with SAP HANA

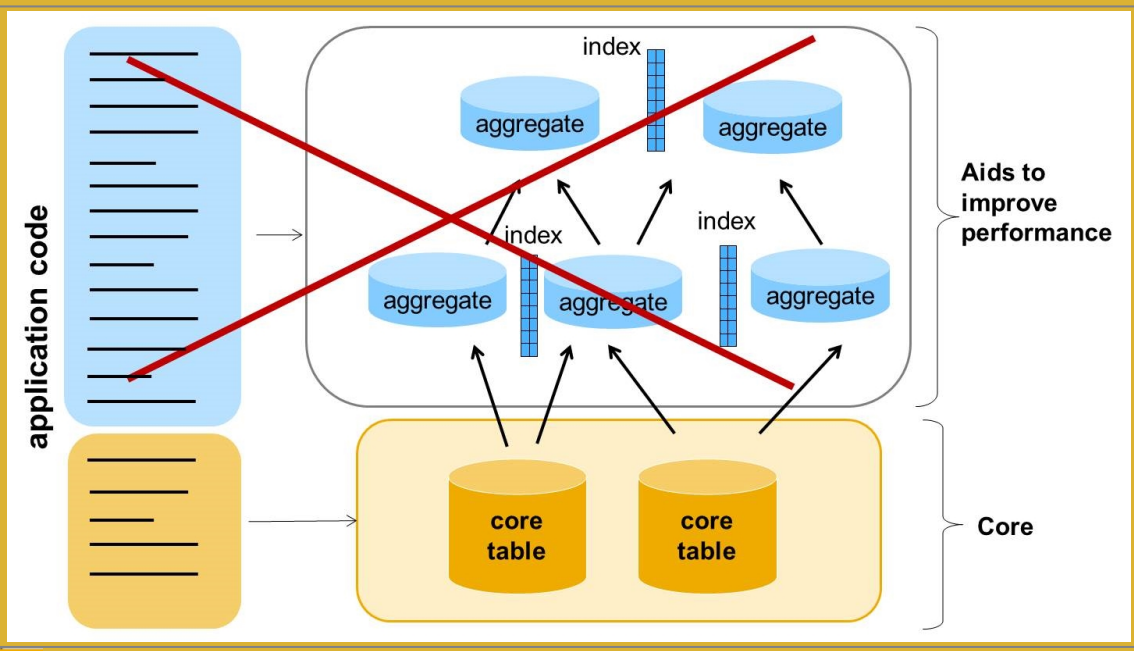


Figure 8: Eliminate Complexity of Applications with SAP HANA

Using the power of SAP HANA, you can aggregate on the fly from any line item table. You do not need prebuilt aggregates. SAP HANA can generate any view of the data at runtime, all from the same source tables.

SAP HANA organizes data using column stores, which means that indexes are not needed. They can still be created but offer little improvement.

As well as removing the aggregates and indexes from the database, you can also remove huge amounts of application code that deals with aggregates and indexes.

You are left with a simplified core data model as well as simplified application code.

It is now much easier to enhance the applications and integrate additional functions.

Movement of Processing to the Data

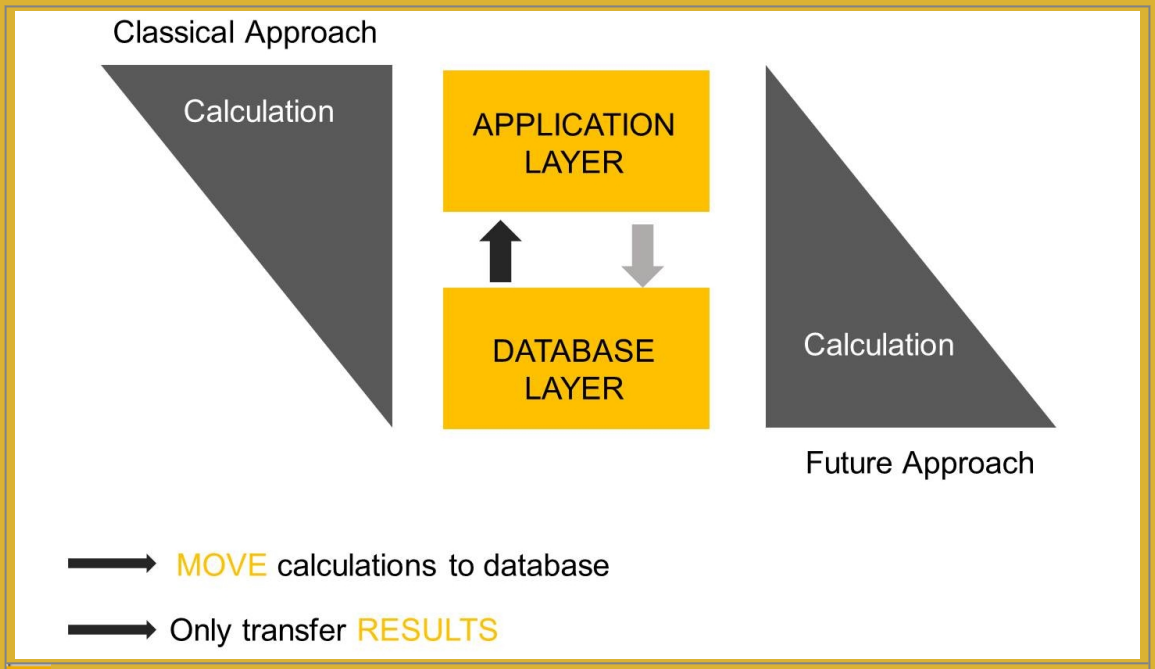


Figure 9: Push Down Processing to SAP HANA

In the past, the role of the database layer was to listen out for instructions from the application layer and act upon these instructions. These instructions could be any of the following:

- Create data
- Read data
- Update data
- Delete data

These requests were simple and required little or no actual processing or calculating of the data. For example, in the case of reading data, the application layer would take care of the data processing tasks once the database had handed over the basic data. The database was told to fetch large blocks of data, or write large blocks of data, and so on. However, SAP HANA is much more than a database, it is a data processing platform.

This means that SAP HANA doesn't just want to be given simple data tasks, it wants to be given all data centric jobs. The more complex the task, the better. These jobs can include the following:

- Aggregate and disaggregate data
- Filter and sort data
- Calculate data
- Convert data
- Cleanse data
- Structure data

Push Down Process

An application should now send all data instructions, whether simple or complex, to SAP HANA. SAP HANA processes the data in memory and sends back only the results. This means that less data passes between the database and application layer.

For example, if the application layer sends the instruction, **`Please summarize the last 5 years' sales line items of yellow widgets into region totals by year, and calculate the net value after discount`** to SAP HANA. Instead of sending millions of basic rows from the database to the application layer, SAP HANA processes the data request and sends back only the results to the application layer. A huge reduction in data volume is being passed. As well as this, the data processing is done in memory by SAP HANA, so performance is excellent too.

Moving the data processing tasks from the application layer to the database layer is called push-down. Push-down means that application developers need to rethink their approach. In the past, all coding focused on the application layer, but now with SAP HANA, large parts of the coding can be pushed down. This means that developers need to think of how to ensure that they pass challenging data processing tasks to SAP HANA, instead of expecting the application layer to handle it.

For simple applications, SAP HANA can take care of all processing and a separate application server is not needed. SAP HANA has a built-in application server called SAP HANA XS, which is covered in detail later in the course.

However, the application layer is still needed with complex enterprise applications such as SAP S/4HANA and Business Warehouse (BW). It is needed to handle the complex business logic that must be programmed in a dedicated business programming language. In the case of SAP applications, the language is ABAP. You cannot develop and run ABAP applications directly in SAP HANA. For that, you need SAP NetWeaver Audience Sensor ABAP, which sits on top of SAP HANA. Now, you have two platforms working as an optimized stack – SAP NetWeaver Audience Sensor ABAP, which provides the application services, running on top of SAP HANA, which provides the data services.

Data Access from Anywhere

In today's digital world there are many types of data. As well as the traditional business data of structured records, we also have the following data types:

- Text Data

  This is data from social media feeds, help desk tickets, logs, and so on.

- Spatial Data

  This is data that relates to locality, maps, engineering diagrams, floor plans, and so on.

- Graph Data

  This is data that relates to highly networked entities such as social networks, supply chains, and so on.

Access Any Type of Data

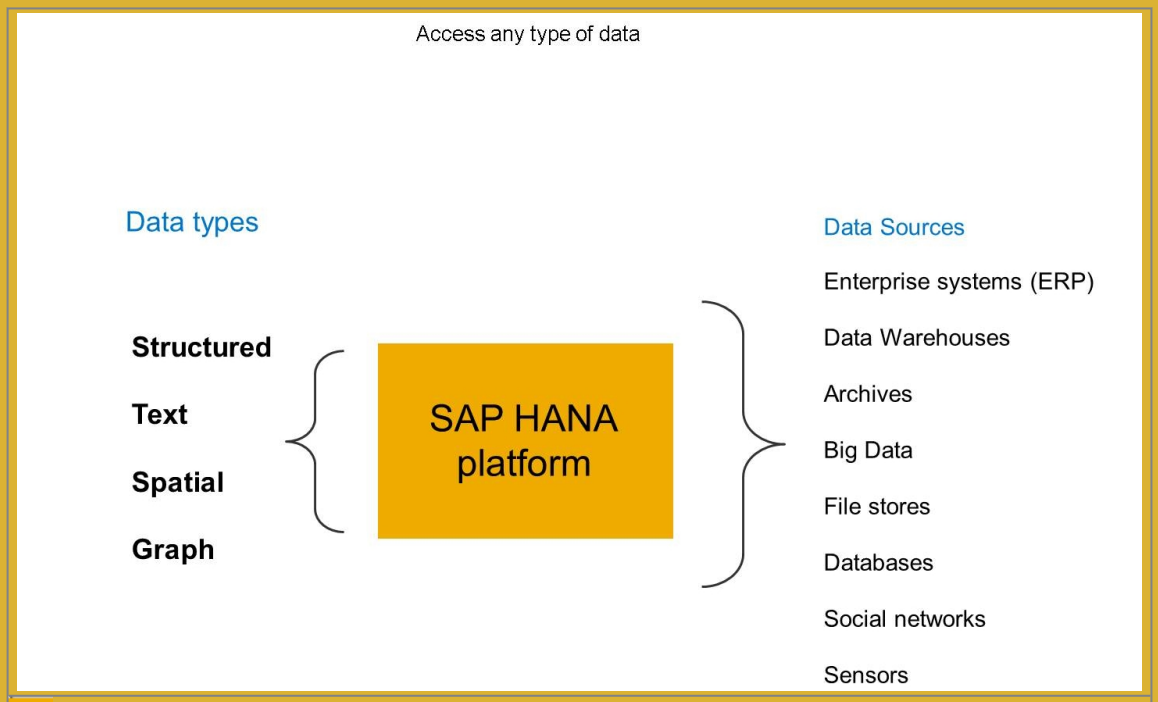


Figure 10: Access Any Type of Data

SAP HANA can process all types of data and also combine them in new and innovative applications. Imagine a cockpit that provides a summary of customer feedback based on an aggregation of social media comments. Imagine being able to drill down on any customer sentiment to see if there is any regional aspect to the sentiment. Finally, imagine drilling down on the map to open up all sales orders for that region related to the original sentiment.

SAP HANA can also access data from any type of source system, including the following:

- Enterprise Systems

  SAP S/4HANA is an example of an enterprise system
- Data Warehouses

  BW is an example of a data warehouse
- Archives

  SAP HANA can create an online connection to remote archives
- Big Data

  Apache Hadoop is an example of Big Data
- File Stores

  XL, CSV, and XML are examples of file stores
- Databases

  This includes any relational database
- Social Networks

  Twitter, Facebook, and LinkedIn are examples of social networks

- Sensors

  These are embedded databases and data containers in smart devices and machines

Access Data Anywhere

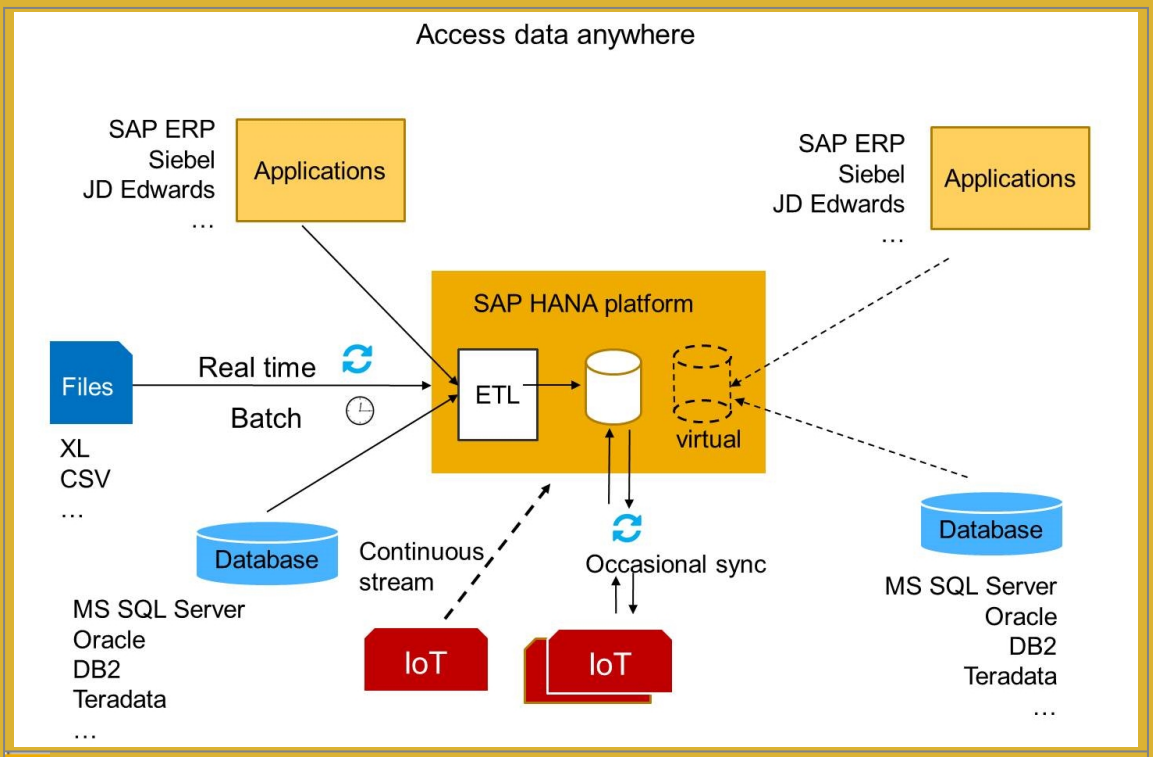


Figure 11: Access Data Anywhere

SAP HANA can consume data for processing from anywhere.

Data can be physically loaded to SAP HANA either in real time or batch. Data can be accessed virtually from remote sources to obtain a live view of data without loading. Data can be loaded from files, databases, continual streams, or occasionally connected sources.

SAP HANA has its own built-in ETL tools. This means that integrating and cleansing data is possible without the need to implement other tools. On the other hand, if external data provisioning tools are required, then SAP HANA can natively connect to those tools.

Platform Services to Power any Type of Application

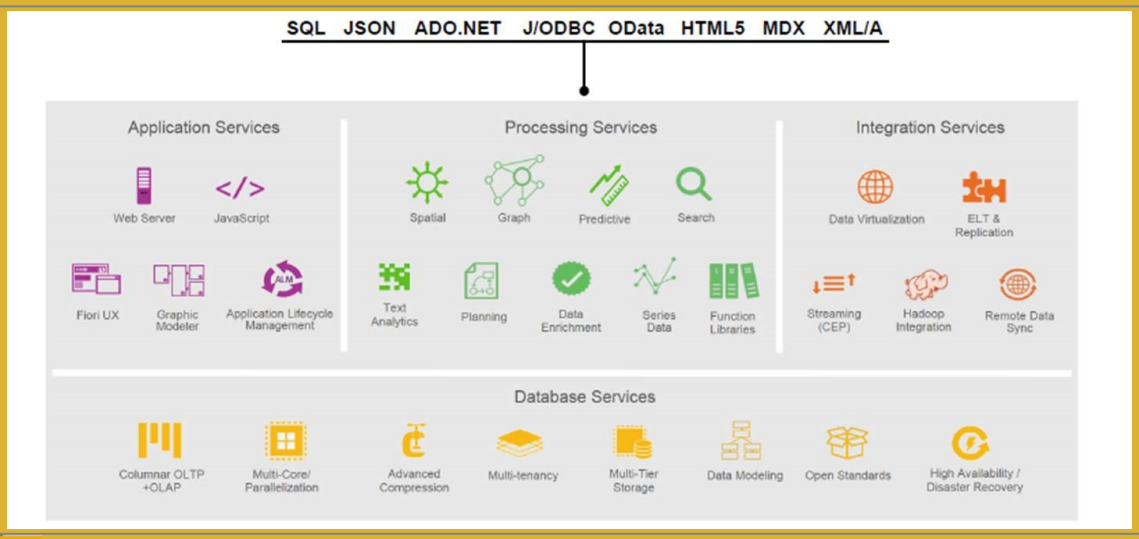


Figure 12: Introducing SAP HANA Platform

You should spend some quality time on this slide, it gives a great run down of the key pieces of SAP HANA. I suggest you tackle this starting with Database Services first (you will always need database services). Have just one sentence ready for each icon to explain the key feature, but don't go into detail as you cover most of this in later lessons. Then move to Processing Services, then Integration Services, then Application Services. That creates a good flow that students can follow and properly describes the huge numbers of services provided by SAP HANA. I would suggest that you give about 15 minutes to this slide, allowing for a few questions. Remember, you cover many of the aspects of this slide in detail later so don't get dragged into detailed discussions.

SAP HANA is a platform. A platform is a combination of many services that power applications. A platform should include all services required by the applications. The platform is the software that was built entirely by SAP only to run on the most powerful hardware.

The SAP HANA platform combines all key components that can be used to power any application. These components include database, data processing, application development tooling, lifecycle management, and data integration. They can be used in both SAP and non-SAP applications.

Service Key Capabilities of SAP HANA

The following are some of the key capabilities of SAP HANA, organized by service:

- Application Services

  As well as a database, SAP HANA also provides many application services. This means many applications are built in a two tier model, rather than a three tier model. For example, imagine an application that allows a project manager to quickly check that all team members have competed their time sheets. This could be developed as a web application where only a web browser and SAP HANA is required, and no application server is needed. This is because SAP HANA can handle the business logic as well as the database services. SAP HANA provides a full development environment with productivity tool supplied in the box. Everything the developer needs at design time and runtime is there.

- Processing Services

  SAP HANA can handle many new types of data. This includes text, spatial, graphic, and more. However, it is not enough to store these new data types. You also need to be able to build applications that can process and integrate this data with traditional data types, such as business transactions. SAP HANA provides native in-memory engines that process all types of data in real time.

- Integration Services

  SAP HANA has the following built-in data consumption options:

  - Continual streaming data analysis
  - Read data remotely in any data source
  - Read from Big Data stores such as Hadoop
  - Synchronizes in both directions with remote databases and devices that collect data (IoT)

  SAP HANA has built-in Extraction, Transformation, and Loading (ETL) capabilities. This means that separate software is no longer needed to clean, enrich, and profile data from any source.

- Database Services

  SAP HANA is a full, in-memory column-and-row store database that can support both OLTP and OLAP requirements and is built to run on high-end hardware. It stores data optimally using automatic compression and is able to manage data on different storage tiers to support data aging strategies. It has built-in high-availability functions that keep the database running and ensure mission-critical applications are never down.

Deployment Options for SAP HANA

SAP HANA can be deployed either on-premise or accessed as a public cloud service.

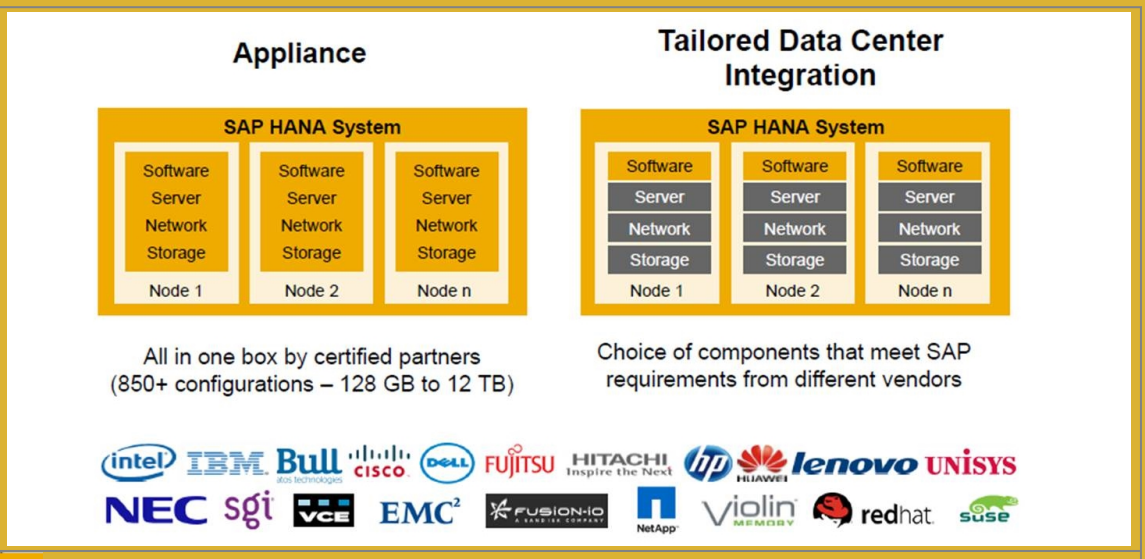


Figure 13: On-Premise Options

For on-premise deployments, SAP HANA can be delivered as an appliance. This means the SAP HANA software is pre-installed on certified hardware. There is a growing list of suppliers who can provide hardware that is certified by SAP to run SAP HANA and these suppliers also pre–install SAP HANA on that hardware. Customers work with the supplier to choose the

correct hardware sizing and configuration. SAP certified suppliers have been trained by SAP to install and configure SAP HANA optimally on their hardware, and their consultants have to pass exams to maintain their certification status.

Alternatively, SAP HANA can be installed by the customer on mixed hardware components that are listed on SAP's approved hardware list. Many customers already have hardware components as well as software licenses that they would like to re-purpose. So, this flexible approach ensures that implementation costs are kept to a minimum and that hardware is recycled. This approach is known as **Tailored Data Center Integration**.

> Note:
> Only certified (C_HANATEC) SAP HANA consultants are allowed to install SAP HANA for production purposes. This restriction does not apply for non-production installations.

Versions of Linux

SAP HANA runs on the Linux operating system. The following versions of Linux are supported:

- SUSE
- Red Hat

CPU architecture choices

For CPU architecture, SAP HANA currently runs on:

- Intel x86
- IBM Power Systems

SAP HANA as a Service (HaaS)

SAP HANA is available as a cloud service. The full name of this service is **SAP Cloud Platform, SAP HANA service** but it is shorten to SAP HANA as a Service (HaaS).

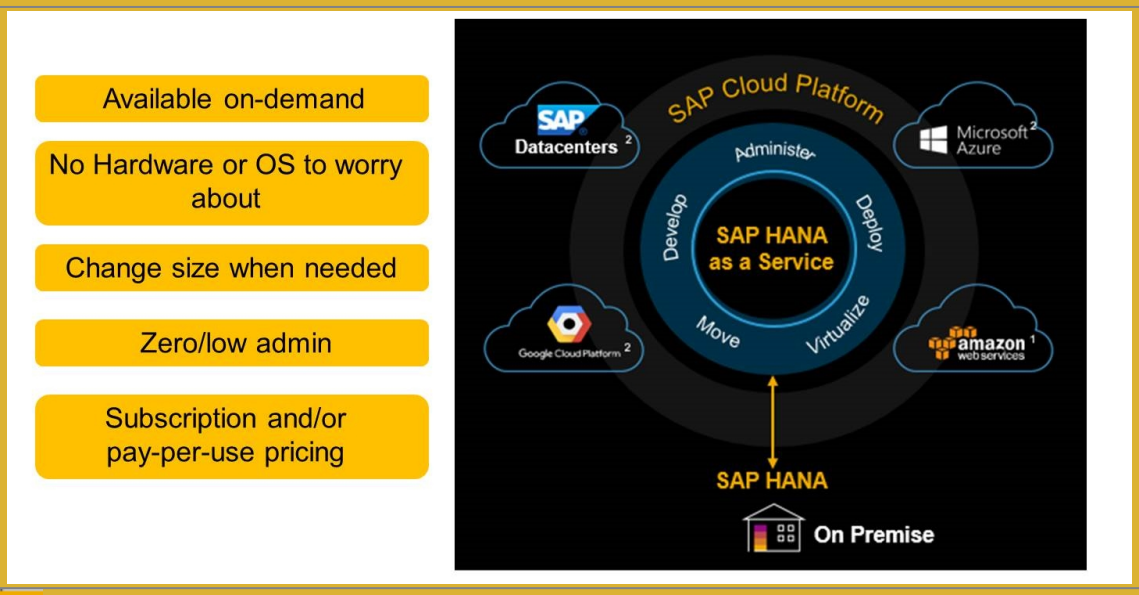


Figure 14: SAP HANA as a Service (HaaS)

SAP HANA as a Service (HaaS) provides a way for customers and partners to use SAP HANA as a managed service in the cloud. It provides improved agility and reduces administration tasks compared to deploying SAP HANA on-premise. This service has been available for a long time and was originally based on Persistence Service Architecture (PSA), but recently SAP introduced a next generation implementation of the service based on the modern, and open **Cloud Foundry** environment. This provides more opportunities to integrate cloud services from non-SAP suppliers into a common platform.

On-Premise vs. Cloud Comparison

On-premise means that the entire SAP HANA solution : the software, network, and hardware is installed and managed by the customer. The advantage of the on-premise option is that customers have complete control over maintenance and upgrades. Another advantage is that customers can fine-tune the performance of their own deployment. The disadvantage is that customers must have their own in-house resources (personnel, skills, facilities), and purchase or lease their own hardware and licenses.

A cloud deployment is operated and managed by SAP or other certified hosting partners. The advantage of the cloud option is that customers do not have to be concerned with providing and managing the infrastructure. They do not need to have their own operation teams to run SAP HANA. They can simply get on with using and developing applications with SAP HANA. It also means customers do not have to purchase hardware and software licenses. With the cloud option, you can change your configuration size anytime, so as you applications grown you can purchase more, or even go smaller if you find you are not using your full capacity. SAP HANA is paid for by subscription or by usage. It is considered to be a service, just like a utility such as gas or water.

The disadvantage with the cloud solution is that SAP or partners control the maintenance and upgrade schedules, so customers have no control over this. Thus, they have to follow SAP's lifecycle, which tends to move forward at a good pace. This might not be ideal if the customer prefers to control their own software lifecycle, especially if the customer is constantly developing add-on applications and needs a more stable platform. Or runs a mission critical application and needs complete control over this including the ability to fine-tune the performance. The cloud solution receives regular updates — about every 2 weeks — and this can be seen as an advantage if you are looking for the very latest innovations on whish to develop applications, or a disadvantage if you are seeking stability with mission critical applications that do not need frequent new features added.

Note:
Be careful not to confuse a public cloud offering with managed cloud. Both of these are cloud solutions. SAP HANA as a Service (HaaS) is a public service shared by many customers. The cloud-service provider supplies SAP HANA, the hardware, and the operating resources. A managed cloud is a customer's own private SAP HANA platform but managed by SAP or hosting partners who provide the hardware and operate the system, but the customer must provide SAP HANA and applications. HANA Enterprise Cloud (HEC) is an example of this where customers' S/4HANA or BW systems are in the cloud but they are private, not shared and they are run by SAP and partners.

Hybrid Approach

Another option is a hybrid approach, where a combination of on-premise and cloud is used. For example, a customer wants to have an on-premise deployment of SAP HANA to run their ERP solution. However, they are also developing new applications that require more complex

SAP HANA services and infrastructure than they currently have in-house. Because of this, they use a cloud version of SAP HANA to run these applications. A key point to remember is that a hybrid solution brings together applications in the cloud and on-premise, and there should be no barriers to developing applications that cross both types of deployment. Since SAP HANA 2.0, it is possible to develop applications that can easily be swapped between on-premise and cloud (and the other way around). These are called multi-target applications (MTA) and requires no changes to code.

Note:
To learn more about cloud deployments of SAP HANA, refer to training course **CLD100**.

SAP HANA Scenarios

SAP HANA is central to SAP's strategy of providing a next-generation digital platform that can power both **existing** and **new** applications either on-premise or cloud or hybrid. These applications can be either SAP or non-SAP applications.

SAP HANA can be used in a large number of different IT scenarios. Let's take a quick look at some examples of where SAP HANA can be used.

Scenarios for SAP HANA

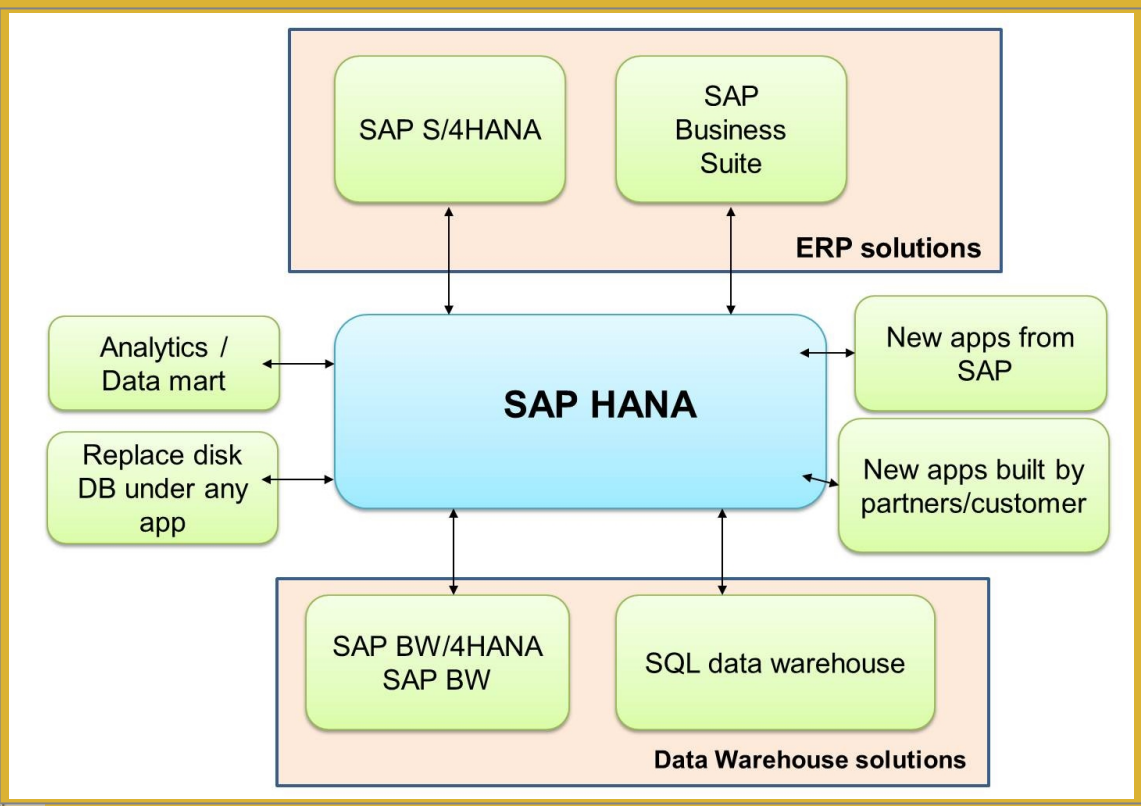


Figure 15: Use Cases for SAP HANA

There are many use cases and scenarios for SAP HANA. These include the following:

- To power enterprise suites such as SAP S/4HANA and SAP Business Suite.
- To replace any relational database.

- To power off-the-shelf enterprise data warehouses such as SAP BW/4HANA or SAP BW, or to build bespoke SQL data warehouses.
- To power partners' and customers' new applications.
- To build local data marts from scratch that can provide real time analytics.
- To power innovative line-of-business SAP packaged applications that are built to run only on SAP HANA, such as Sports Analytics or Smart Meter Analytics.

We cover each of these use cases in more detail in a later lesson.

SAP HANA Versions

SAP HANA is available in two versions; SAP HANA 1.0 (launched in 2011), and SAP HANA 2.0 (launched in 2016). Although you may assume that SAP HANA 1.0 is now out of date, in fact both versions are relevant and each has its own use case.

Since the launch of SAP HANA 1.0, the product has evolved at a phenomenal pace with new features and capabilities being added continuously. SAP has been keen to make available the new features as soon as possible. So SAP delivered updates to SAP HANA through support packs stacks (SPS) every six months. However, customers who run mission critical applications (for example, SAP S/4HANA) prefer an SAP HANA platform with less frequent updates, thus avoiding working through frequent upgrade projects. They may also not require the new functionality delivered in each support pack and thus are not motivated to upgrade. However, in parallel, those customers may also be developing new applications and would like to take advantage of the latest innovations. So SAP decided to offer a dual-track approach to satisfy both needs.

SAP HANA: Versions

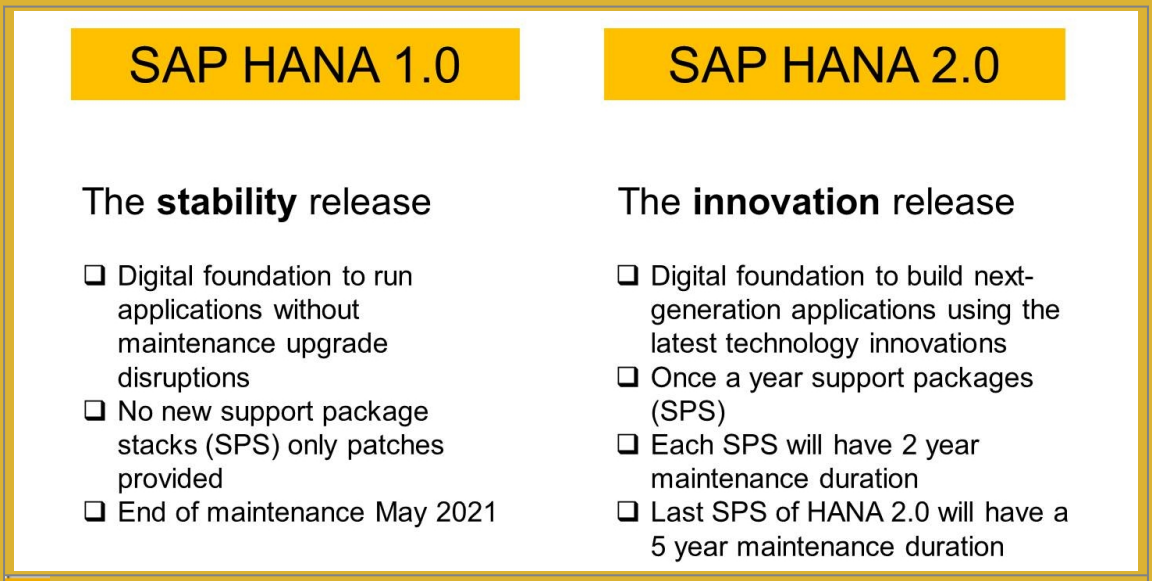


Figure 16: Two Versions of SAP HANA

SAP HANA 1.0 SPS12 is ideal for customers seeking a stable platform without frequent updates to run their mission-critical applications. There is no support packs after SPS12 on SAP HANA 1.0. Only software fixes are provided, through maintenance revisions, until May 2021. Customers running SAP HANA 1.0 should consider upgrading to SPS12 if they are on lower support packs, to get the maintenance until May 2021.

SAP HANA 2.0 is ideal for customers seeking a platform that enables them to leverage the latest technology innovations. Customers can build next-generation applications and also run

mission-critical applications on SAP HANA 2.0. The SAP HANA 2.0 platform inherits all the features and capabilities of SAP HANA 1.0 but with significant additional innovations. All content built with SAP HANA 1.0 continues to run on SAP HANA 2.0 with no changes.

Upgrade Paths for Existing Customers

Customers running SAP HANA 1.0 SPS10 – SPS12 can upgrade directly to SAP HANA 2.0. However, SAP recommends that those customers first take advantage of the new Capture and Replay tool, introduced for SPS12. This tool allows you to capture existing workloads from SPS12 and see how they run on SAP HANA 2.0 This can help to identify any configuration issues, allowing customers to fine tune the target SAP HANA 2.0 system before committing to the changeover.

Customers running SAP HANA SPS09 and lower must first upgrade to SAP HANA 1.0 SPS12 to then upgrade to SAP HANA 2.0.

Unit 1
Exercise 1

Set Up the Training Environment

25

The purpose of this exercise is to set up the training environment so that all future exercises in this course can be executed. Some exercises require ready-made files. You will copy these files to your local work area by running a prepared script so they become available to you.

1. Log on to your training landscape.

> Note:
> The log on procedure to the training landscape will be provided by your instructor or included in the SAP Live Access instructions.

2. (Live Access users only) Connect to your own Live Access system as indicated in the SAP Live Access general user guide.
3. (Classroom Training, Virtual Live Classroom only) Start a remote desktop connection to the training landscape using the following data:

   Table 1: Connect to Remote Desktop

Field	Value
Computer	<provided by your instructor >
User Name	**train-##**
Password	**initial**

   If a dialog box displays the message, *The identity of the remote computer cannot be verified...*, select the checkbox *Don't ask me again* and choose *Yes*.

4. Run the *HA100* initialize script so that the course folder *HA100* and its contents is copied to your local environment, in *N:\HA100*.
5. Add the course folder *N:\HA100* to your *Favorites* so it is easy to access in future.
6. Check that the course resources are present.

Set Up the Training Environment

26

The purpose of this exercise is to set up the training environment so that all future exercises in this course can be executed. Some exercises require ready-made files. You will copy these files to your local work area by running a prepared script so they become available to you.

1. Log on to your training landscape.

> Note:
> The log on procedure to the training landscape will be provided by your instructor or included in the SAP Live Access instructions.

2. (Live Access users only) Connect to your own Live Access system as indicated in the SAP Live Access general user guide.
3. (Classroom Training, Virtual Live Classroom only) Start a remote desktop connection to the training landscape using the following data:

Table 1: Connect to Remote Desktop

Field	Value
Computer	<provided by your instructor >
User Name	**train-##**
Password	**initial**

If a dialog box displays the message, *The identity of the remote computer cannot be verified...*, select the checkbox *Don't ask me again* and choose *Yes*.

a) On the SAP Training Landscape portal or Common Training, choose the *Remote Desktop Connection* icon.

b) In the *Remote Desktop Connection* dialog box, in the *Computer* field, enter the Remote Computer name provided by your instructor and choose *Connect*.

c) In the *Windows Security* dialog box, choose *Use another account* or for Windows 10, select more choices > *Use a different account*..

d) Enter the user name and password from the table, Connect to Remote Desktop, and choose *OK*.

4. Run the *HA100* initialize script so that the course folder *HA100* and its contents is copied to your local environment, in *N:\HA100*.

a) In the Remote Desktop environment, choose the *Start* button.

b) Select the *Initialize Course* tile.

c) Double-click the *HA100* folder.

d) To execute the script, double-click the *Initialize_HA100* shortcut.

e) At the next prompt, choose *Yes* to execute the script.

5. Add the course folder *N:\HA100* to your *Favorites* so it is easy to access in future.
   a) To create a link, drag the *HA100* folder from the local *N:* drive to *Favorites*.
6. Check that the course resources are present.
   a) In *Favorites*, choose the *HA100* folder and ensure that various ready-made files and shortcuts are present.

LESSON SUMMARY

You should now be able to:

- Describe how SAP HANA powers a digital platform

Learning Assessment

29

1. Which recent trends have triggered the need for a next generation data processing platform?

   Choose the correct answers.

   - [X] A Increase in connectivity of people and devices
   - [X] B Separation of transactional processing and analysis processing to dedicated servers
   - [X] C Increase in use of mobile devices
   - [X] D Massive growth in data volume

2. Which recent technology innovations have triggered the opportunity to build a next generation data processing platform?

   Choose the correct answers.

   - [X] A Faster disk access
   - [X] B Multi-core processing
   - [X] C Larger memory availability

3. What are key features of SAP HANA?

   Choose the correct answers.

   - [X] A Fully in-memory database
   - [X] B Automatic compression of data
   - [X] C Cache not needed

4. Why might customers choose to implement SAP HANA as a Service (HaaS)?

   Choose the correct answers.

   - [X] A They prefer a simpler version of SAP HANA
   - [X] B Reduce administration effort on customer side
   - [X] C Pay per usage
   - [X] D Change size as needed
   - [X] E They prefer to re-use their own hardware components

5. What is meant by SAP HANA 'Tailored Data Centre Integration'?

   Choose the correct answer.

   - [X] A Customer creates a customized hardware platform on which to install SAP HANA, by reusing existing and SAP certified hardware components
   - [X] B Partners provide cloud infrastructure on which SAP HANA can be installed
   - [X] C Partners provide a fully installed SAP HANA on certified hardware for use on-premise

Learning Assessment - Answers

31

1. Which recent trends have triggered the need for a next generation data processing platform?

   Choose the correct answers.

   - [X] A Increase in connectivity of people and devices
   - [X] B Separation of transactional processing and analysis processing to dedicated servers
   - [X] C Increase in use of mobile devices
   - [X] D Massive growth in data volume

   Correct!

2. Which recent technology innovations have triggered the opportunity to build a next generation data processing platform?

   Choose the correct answers.

   - [X] A Faster disk access
   - [X] B Multi-core processing
   - [X] C Larger memory availability

   Correct!

3. What are key features of SAP HANA?

   Choose the correct answers.

   - [X] A Fully in-memory database
   - [X] B Automatic compression of data
   - [X] C Cache not needed

   Correct!

4. Why might customers choose to implement SAP HANA as a Service (HaaS)?

   Choose the correct answers.

   - [X] A They prefer a simpler version of SAP HANA
   - [X] B Reduce administration effort on customer side
   - [X] C Pay per usage
   - [X] D Change size as needed
   - [X] E They prefer to re-use their own hardware components

   Correct!

5. What is meant by SAP HANA 'Tailored Data Centre Integration'?

   Choose the correct answer.

   - [X] A Customer creates a customized hardware platform on which to install SAP HANA, by reusing existing and SAP certified hardware components
   - [X] B Partners provide cloud infrastructure on which SAP HANA can be installed
   - [X] C Partners provide a fully installed SAP HANA on certified hardware for use on-premise

   Correct!

UNIT 2 Architecture of SAP HANA

UNIT OBJECTIVES

- Outline components of SAP HANA
- Work with SAP HANA interfaces
- Understand key features of SAP HANA database
- Describe high availability

Outlining Components of SAP HANA

35

LESSON OVERVIEW
This lesson provides an overview of the key components in an SAP HANA landscape.

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Outline components of SAP HANA

Core SAP HANA Components

An SAP HANA landscape is made up of many software components. Not all components are required by customers and so rather than delivering one super-sized version of SAP HANA containing all components – therefore requiring more hardware and resources – SAP provides different **editions** of SAP HANA. It might help to think of editions as 'bundles' of software components. The current editions for SAP HANA 2.0 at the time of creating this course (summer 2018), are as follows:

SAP HANA, Enterprise edition
SAP HANA, Standard edition
SAP HANA, Express edition

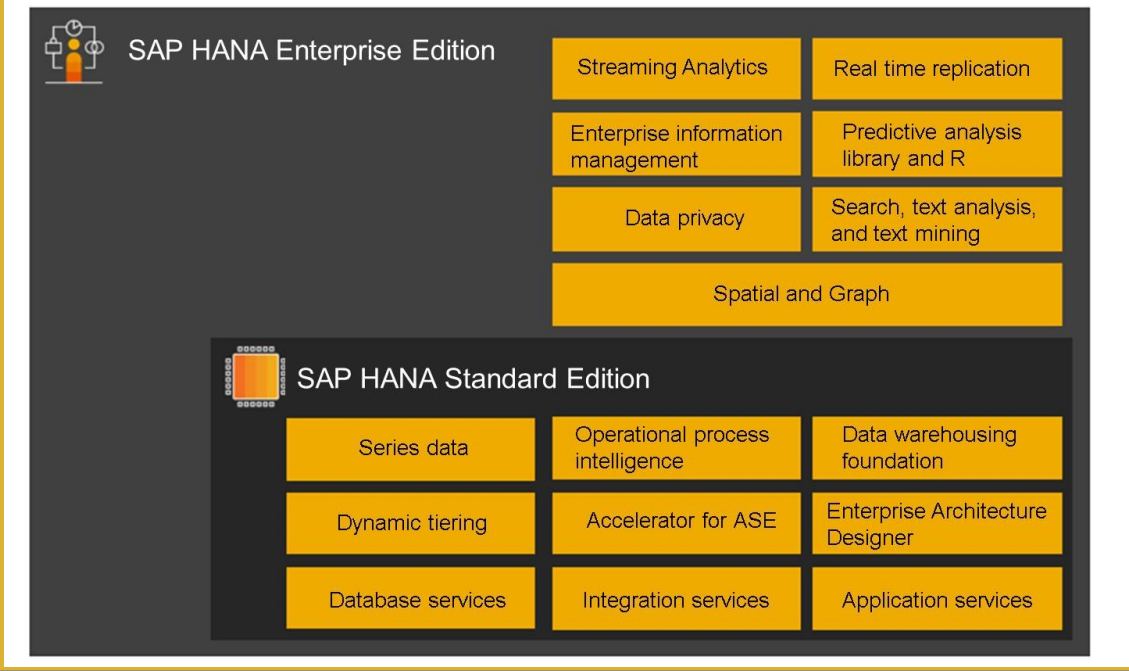


Figure 17: Editions of SAP HANA

The SAP HANA Standard edition comprises the core database, integration and application services, series data, plus a number of components that require additional installation.

The SAP HANA Enterprise edition comprises everything in the Standard edition plus extra components to power sophisticated applications.

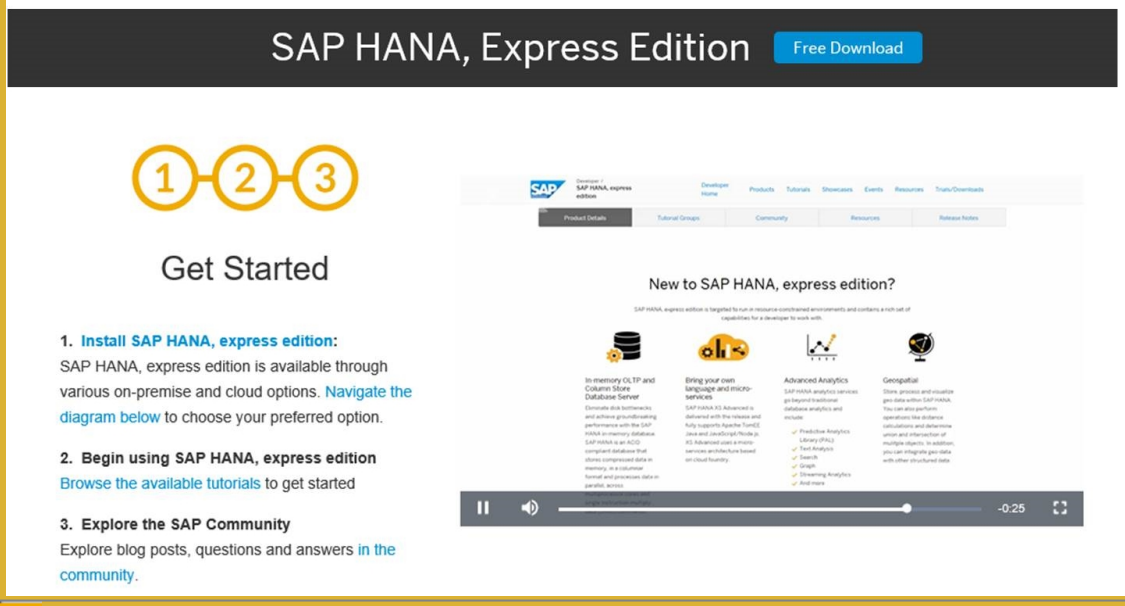


Figure 18: SAP HANA Express Edition

The SAP HANA Express edition is a free-of-charge edition aimed at developers who would like to build and deploy applications on SAP HANA. This edition does not have the same hardware sizing requirements of the Standard and Enterprise editions, and can be installed on a laptop or desktop machine with a reasonable specification. It is easily downloaded and is supported by the SAP Community. If you do not have the required hardware specification then you can still deploy SAP HANA Express edition in a certified cloud provider. With the SAP HANA Express edition you can build and deploy applications that use 32GB of memory without charge. You can easily expand by purchasing more memory usage up to 128GB. To move beyond 128GB or to add the advanced processing options such as streaming analytics, you can upgrade to a full-use licence edition such as Standard or Enterprise editions.

Note:
You can find out more about the latest editions and what they contain by going to the SAP Help Portal and looking up the **Feature Scope Description for SAP HANA**.

In this course we don't focus on editions but describe SAP HANA without such constraints.

Careful: Stay away from discussions about what is included in each SAP HANA Edition and what the Options are at each release. Let the students find out for themselves by using the SAP documentation mentioned previously. The features delivered in Editions and Options often changes with each release. This can end up getting confusing for everyone and we should not get into commercial/license discussions in the classroom. Also, if needed, refer students to this excellent blog, which covers this topic extremely well, and also goes back to describe editions and options from the very beginning when SAP HANA was introduced. Emphasize that they should try to find out which editions and options they have/or will have. But also encourage everyone to keep an open mind and learn about all of the capabilities of

SAP HANA and not get hung up what is limited by editions: The blog is found at https://blogs.sap.com/2017/02/01/sap-hana-2.0-editions-and-options-by-the-sap-hana-academy and they do keep it up to date.

SAP HANA Basic Landscape

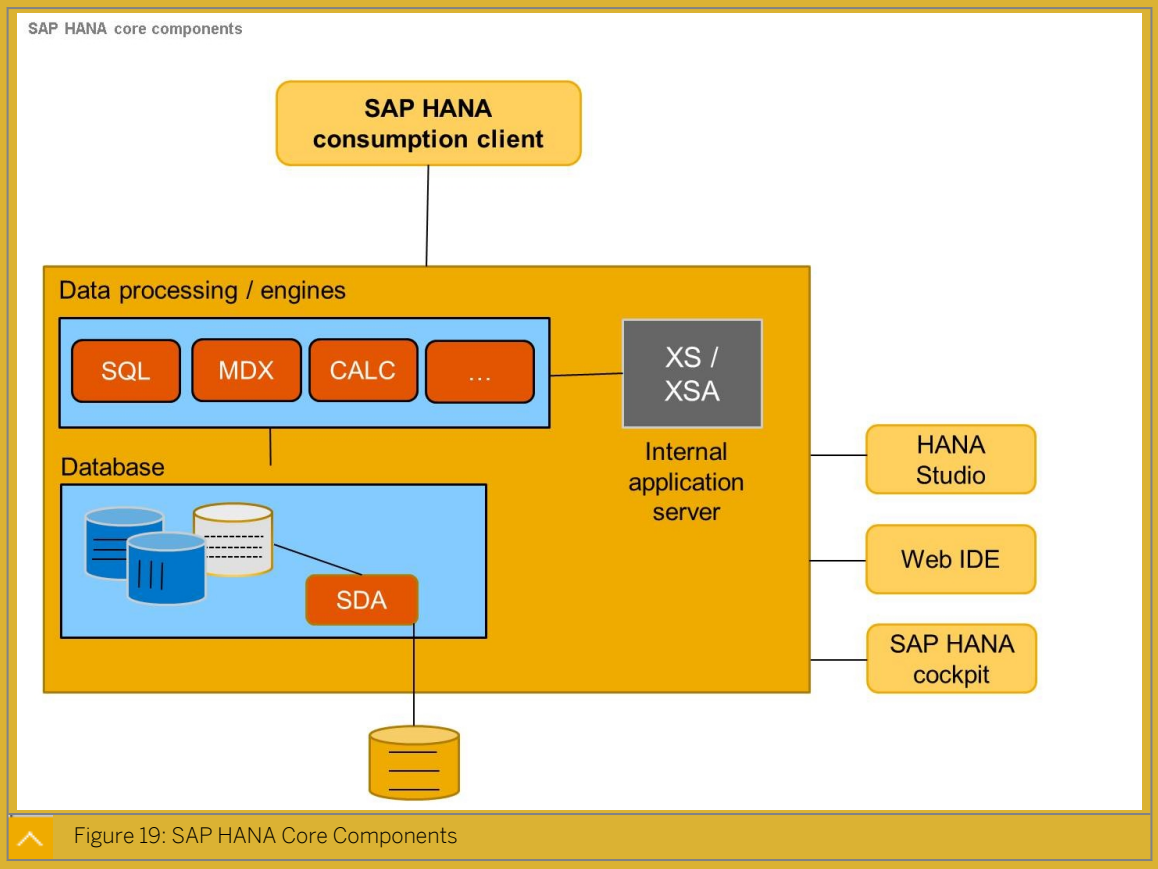


Figure 19: SAP HANA Core Components

At the heart of SAP HANA is the in-memory database. The SAP HANA database uses row and column store tables. The database is a fundamental component of SAP HANA and therefore it is a mandatory part of the landscape.

The database tables contain physical data. There can also be virtual tables that are connected to external data sources, which provide live data on request. A virtual table is a logical table that contains no data in SAP HANA but it is treated just like a regular table. The Smart Data Access (SDA) component performs the provision of live data to the virtual tables.

Data Processing Engines

Another set of essential components of SAP HANA are the data processing engines. These dedicated engines provide the in-memory number crunching capabilities. They listen out for requests from applications and then interact closely with the database to produce fast results within the database. There are many engines available, each specializing in handling different types of data. But the engines work together, and special optimizers figure out which engine should be called to get the best performance.

SAP HANA provides relational access to the database via SQL and multidimensional access via MDX. There are dedicated processors to handle these query requests. The calculation engine is able to undertake complex analytical tasks, such as hierarchical and deep OLAP queries. It works in harmony with SQL and MDX processors to produce the fastest possible

results. There are many processing engines in SAP HANA, including text processing engines, spatial engines and graph engines all working together to provide optimal performance.

XS/XSA

Extended Application Services (XS) and its more powerful replacement, Extended Application Services – Advanced (XSA) are the two built-in application servers that are part of core SAP HANA. With XS or XSA, you have all the development and runtime tools needed to develop and run your own native SAP HANA applications with no separate application server required (for example, NetWeaver). All you need to add is a front-end client such as a browser and you have the complete end-to-end application stack.

Interfaces

The SAP HANA Studio is a thick client (runs on Windows, MacOS, or Linux) and provides the front-end interface that serves the needs of multiple roles including developers, data modelers, and administrators. These people log on directly to SAP HANA to access the development tools or perform administration duties. With SAP HANA Studio, you can only develop XS based applications (not XSA).

The Web IDE for SAP HANA is the next generation web interface used to develop XSA based applications and data models. It cannot be used for XS development. The Web IDE for SAP HANA contains no administration tools, it is purely a tool for development.

The SAP HANA Cockpit is used for administration and monitoring of the SAP HANA platform and is based on SAP Fiori.

Extended SAP HANA Landscape

As well as the core components, there are many additional components that make up an SAP HANA landscape. Customers choose these additional components based on their needs.

Note:
This is not a complete list of all possible additional components, but is meant to provide an illustration of the type of components that are available.

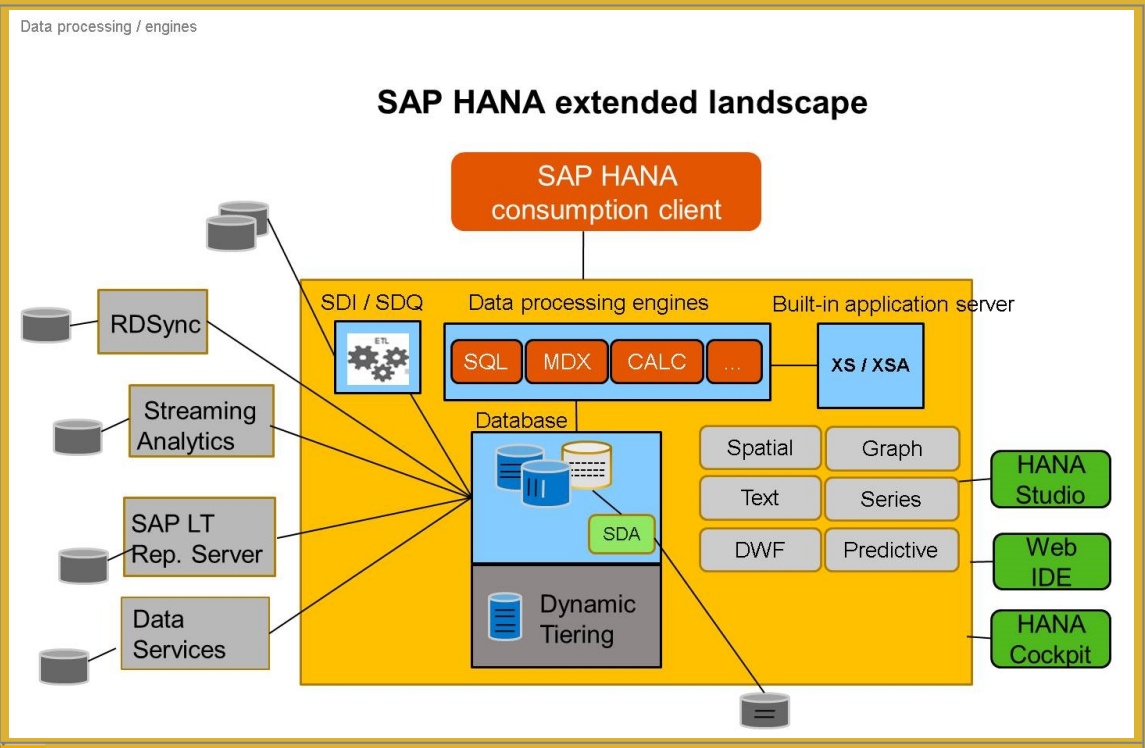


Figure 20: SAP HANA Additional Components

SDI/SDQ

SAP HANA has its own built-in Enterprise Information Management (EIM) component that takes care of the majority of data provisioning scenarios. This is used to provision data from any sources to SAP HANA in real-time or batch. EIM consists of a component called Smart Data Integration (SDI). SDI takes care of extraction, transformation, and loading (ETL) of data. The other component of EIM is Smart Data Quality (SDQ). You use SDQ to enrich and cleanse data during the provisioning process.

SAP Landscape Transformation Replication Server and SAP Data Services

SAP HANA natively connects to other SAP data provisioning tools, such as SAP Landscape Transformation Replication Server (SAP LT Replication Server) and SAP Data Services. These optional tools are used to acquire and transform data from SAP and non-SAP sources in real-time or batch. Many customer already use these tools, and so SAP have ensured that they are able to work natively with SAP HANA. However, SAP recommends the use of the built-in EIM component in order to simplify the SAP HANA landscape, so these external tools can be mostly eliminated.

Streaming Analytics and Remote Data Sync

Streaming Analytics (previously known as SDS) is a component that allows you to connect SAP HANA to live continual multi-channel streams of data. These could be from connected machines or sensors or logs.

RDSync is the remote data synchronization component that is used to connect to enterprise databases and mobile device databases that cannot be online continuously. It provides two-way synchronization services.

Advanced Data Processing

Text processing is an additional component that can be added to provide text processing and analysis services, such as text search and text mining.

Spatial processing is another additional component that can be added to provide analysis services. These are based on spatial data, such as queries on locality, distance, and so on.

Graph processing provides the definition and querying of data that is best described as highly networked, such as a social media friends group.

Series processing allows you to analyze data that is organized into measurable time intervals such as gas meter readings so that you can perform queries on this data to identify patterns, and so on.

Predictive processing supports deep analysis of data patterns in order to provide insights and probable outcomes.

Data Warehousing Foundation (DWF)

Data Warehouse Foundation (DWF) provides the extra functionality and components that are required when building a native data warehouse using only SAP HANA (not BW).

Dynamic Tiering

Dynamic Tiering provides a disk-based tier to complement the existing memory tier of the SAP HANA database. This means that you can offload less critical data to a disk-based store, thus freeing up memory for only the most important data. The disk based-data is still fully accessible to any application just as if it were in memory. However, performance is affected. This is often acceptable for older data or less frequently used data where fast database responses are not essential.

SAP Enterprise Architecture Designer

Introducing SAP Enterprise Architecture Designer, edition for SAP HANA

SAP Enterprise Architecture Designer, edition for SAP HANA (SAP EA Designer) is used to support the **planning** and **design** phase of an SAP HANA project.

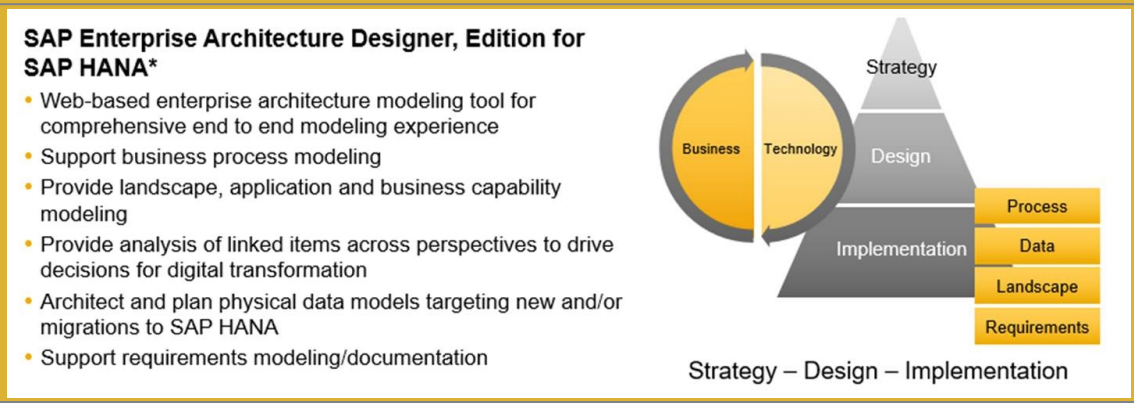


Figure 21: SAP Enterprise Architecture Designer, Edition for SAP HANA

SAP EA Designer is a tool that was introduced with SAP HANA 2.0 and is shipped with SAP HANA and includes five user licenses to get you started. The origins of SAP EA Designer can be traced back to the industry leading enterprise planning and design tool, PowerDesigner, which was originally developed by Sybase. SAP EA Designer is a rewritten version of PowerDesigner built on the SAP HANA platform, with a web only interface and includes the key features of PowerDesigner.

Note:
PowerDesigner is not replaced by SAP EA Designer. PowerDesigner has a large customer base and continues to be an active product with many powerful features not available in SAP EA Designer. SAP EA Designer makes sense for customer who are building SAP HANA applications as it fully integrates with the target SAP HANA system you are building and automates many of the building tasks.

So what can you do with SAP EA Designer?

SAP EA Designer lets you design your organization's landscapes, strategies, requirements, processes, data flows, and data models in a collaborative web-based tool. You create documents to capture each aspect of the project and all documents are fully integrated with each other. This means you can navigate and drill down to all areas of details a project. SAP EA Designer is a single tool that can replace the various planning tools that organizations currently used to provide a 360 degree view of a project.

.

So instead of diving straight into SAP HANA to create connections, data flows and physical schemas, tables, views, and so on, you should first develop the basic design in SAP EA Designer. You can then share the design with your teams and all stakeholders to consider the impact, and collaborate to improve the design. They can add comments, or alternative designs that can then be considered before committing to the final blueprint.

One of the most useful features of SAP EA Designer is the ability to automatically generate physical data modes from logical models. This means that once the design is agreed, the tool can automatically generate the physical objects in SAP HANA such as tables, calculation views and so on. SAP EA Designer cannot generate the complete physical model but it can certainly generate the basic shape so that you can then use the SAP HANA tools, such as Web IDE to fill in the details.

SAP EA Designer can also reverse engineer a physical model sourced from SAP HANA or other well known databases, to generate an SAP EA Designer document to form the logical model. From the logical model, you can then generate a physical model to SAP HANA. This provides a powerful tool used in database migration when moving to SAP HANA from legacy databases.

Note:
SAP EA Designer is available on-premise or as a cloud service.

LESSON SUMMARY
You should now be able to:

- Outline components of SAP HANA

Unit 2
Lesson 2

Working with SAP HANA interfaces

42

LESSON OVERVIEW
This lesson covers the interfaces used by administrators and developers.

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Work with SAP HANA interfaces

SAP HANA Studio

SAP HANA Studio was the original interface for developers and administrators and is still used today, especially for customers running SAP HANA 1.0. SAP HANA Studio can also be used with SAP HANA 2.0 when working in the XS classic development mode. However, SAP HANA Studio cannot be used to develop XSA applications as it does not work with the new HDI infrastructure. SAP HANA Studio is a powerful multi-purpose interface used by several user roles including developers, administrators, and modelers. SAP HANA Studio is a Java application that runs on Windows, Apple MacOS, and Linux. It is based on the well-known, open-source Eclipse and it includes many add-ins provided by SAP to support SAP HANA.

Although we have a new web based interface — Web IDE for SAP HANA — SAP HANA Studio remains relevant today as there are still customers who use SAP HANA 1.0. Also there are still SAP applications that are powered by SAP HANA 1.0 such as SAP Business Suite, it is important to remember that since SAP HANA 2.0, the new interface, SAP Web IDE for SAP HANA will get all new functionality developed by SAP. SAP HANA Studio is no longer developed and its use will reduce over time until it is completely unnecessary. So for developers and modelers who used Studio, they can now use Web IDE for SAP HANA and for administrators who used SAP HANA Studio they can now use another powerful web based interface called SAP HANA Cockpit.

Be sure to make it clear that SAP HANA Studio is still relevant, especially for SAP HANA 1.0. As this course covers both HANA 1.0 and 2.0 customers, we want to include it for completeness. The key thing is not to appear to be showing 'old' software, especially so early on in the class. Some students may question why we still need to show Studio. Studio is very active (at least for now) and especially consultants will be moving between releases of HANA across their customers' projects so they should learn both interfaces. A brand new customer will never care about Studio.

Introducing Connections

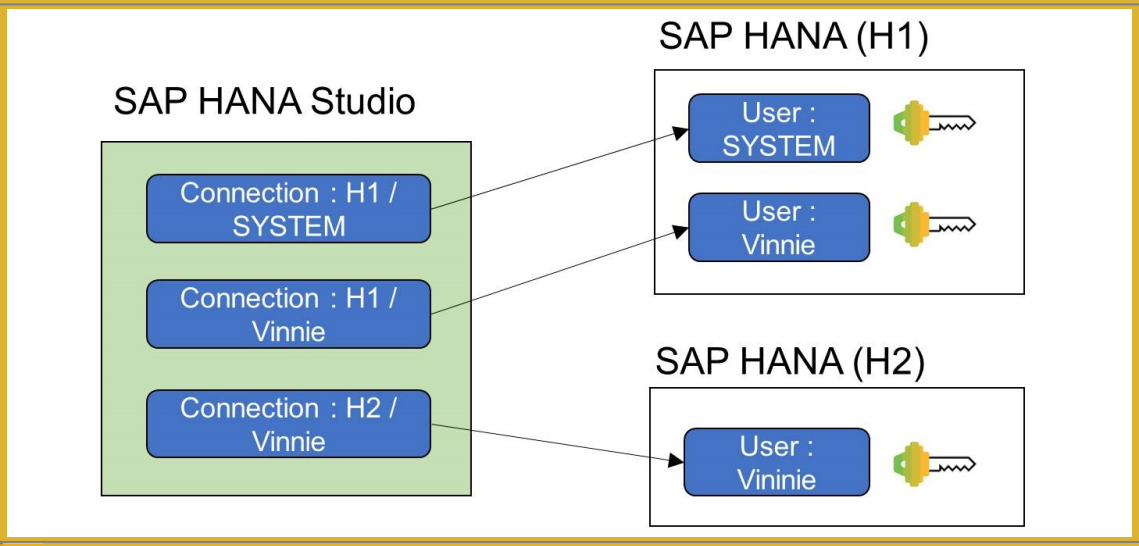


Figure 22: Creating a Connection with SAP HANA Studio

To get started with SAP HANA Studio, you must first create a **connection** to the SAP HANA target system that you want to work with.

A connection is a link between SAP HANA Studio and the SAP HANA platform. After opening SAP HANA Studio, you can create as many connections as you wish to point to different SAP HANA Platforms. This means that you can open the SAP HANA Studio once and then navigate between SAP HANA platforms easily, and all on one screen.

You can optionally give each connection a description so it is easy to identify the purpose of each system when the list of connections becomes long. For example, to identify TEST, DEV, PROD, QA, and so on.

When defining the connection, you are asked to specify the following important information:

- The host and instance (identifies the exact SAP HANA target system).
- A valid SAP HANA database user and password that has been given sufficient privileges on the target SAP HANA system for you to access what you need.

When you log off and then back on to the SAP HANA Studio where connections already exist, you need to provide the passwords for each connection again in order to use them.

It is possible to export the list of connections to a file, so these can then be imported by others. Doing this would mean that they do not have to manually define the connections. Of course, the user credentials are not saved and need to be re-entered.

You can also use the exported list of connections and share them as a central store. Each user creates a link to this central store, and thus does not need to create their own connections or import connections. This means that all connection information is managed centrally, so any changes are made in just one place.

Introducing Perspectives and Views

When you start the SAP HANA Studio for the first time, you are presented with the *Welcome* page. From here you select a **perspective**. Perspectives are simply predefined layouts that contain several panes, or **views** as they are referred to in SAP HANA Studio. One or more perspectives address the needs of a particular SAP HANA user role. For example, a System Administrator uses the *SAP HANA Administration Console* perspective; a BI person might use

the *SAP HANA Modeler* perspective. You can switch between the perspectives as you wish to access the features you need. You can even create a custom perspective so you can design your own layout to suit your needs.

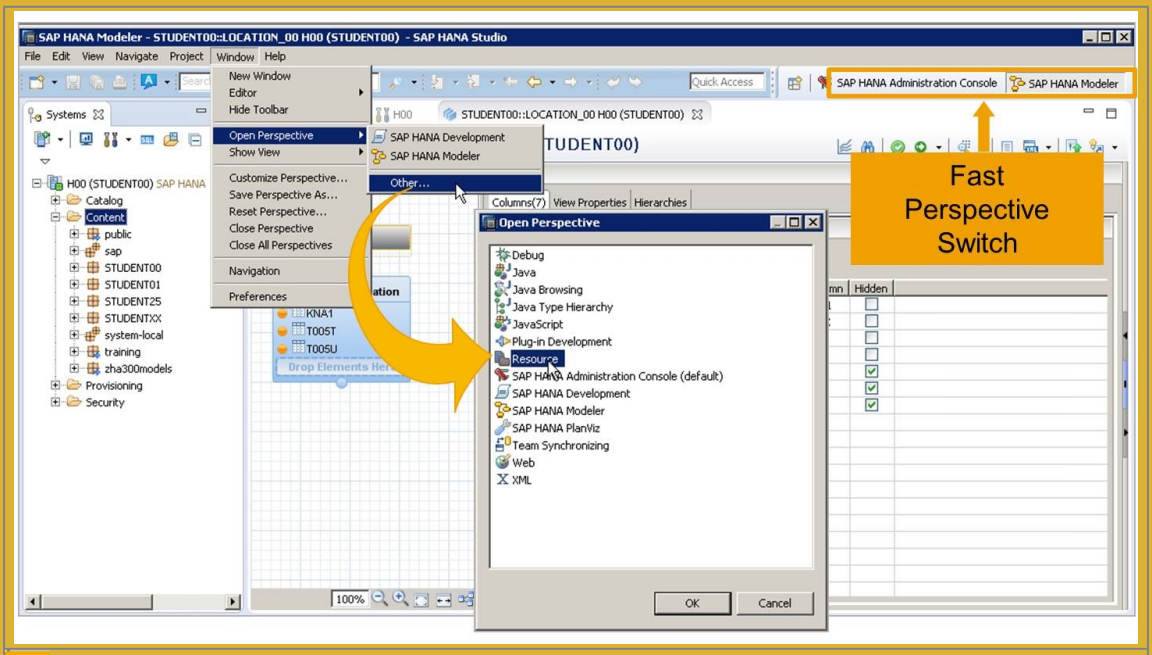


Figure 23: Opening a Perspective

The following are examples of some perspectives that are available in the SAP HANA Studio:

- *SAP HANA Modeler*

  The *SAP HANA Modeler* perspective is used by Data Modelers to create Information Models, as a combination of attributes, dimensions, and measures, included in different types of modeling views.

- *Administration Console*

  The *Administration Console* perspective is used by SAP HANA Administrators to administrate and monitor the whole SAP HANA system.

- *Development*

  The *Development* perspective is used by application developers (coders) to build native SAP HANA applications.

A **view** is simply an individual pane, within a perspective that provides specific information, such as a *Where Used* list or *Properties*. Each view can be moved around via drag and drop. You can also customize the perspectives by adding or removing views. Views can appear in multiple perspectives. For example, the *System* view is used in most perspectives as it presents a hierarchical list of objects in each SAP HANA system that is useful to every role.

Perspectives Based on Views

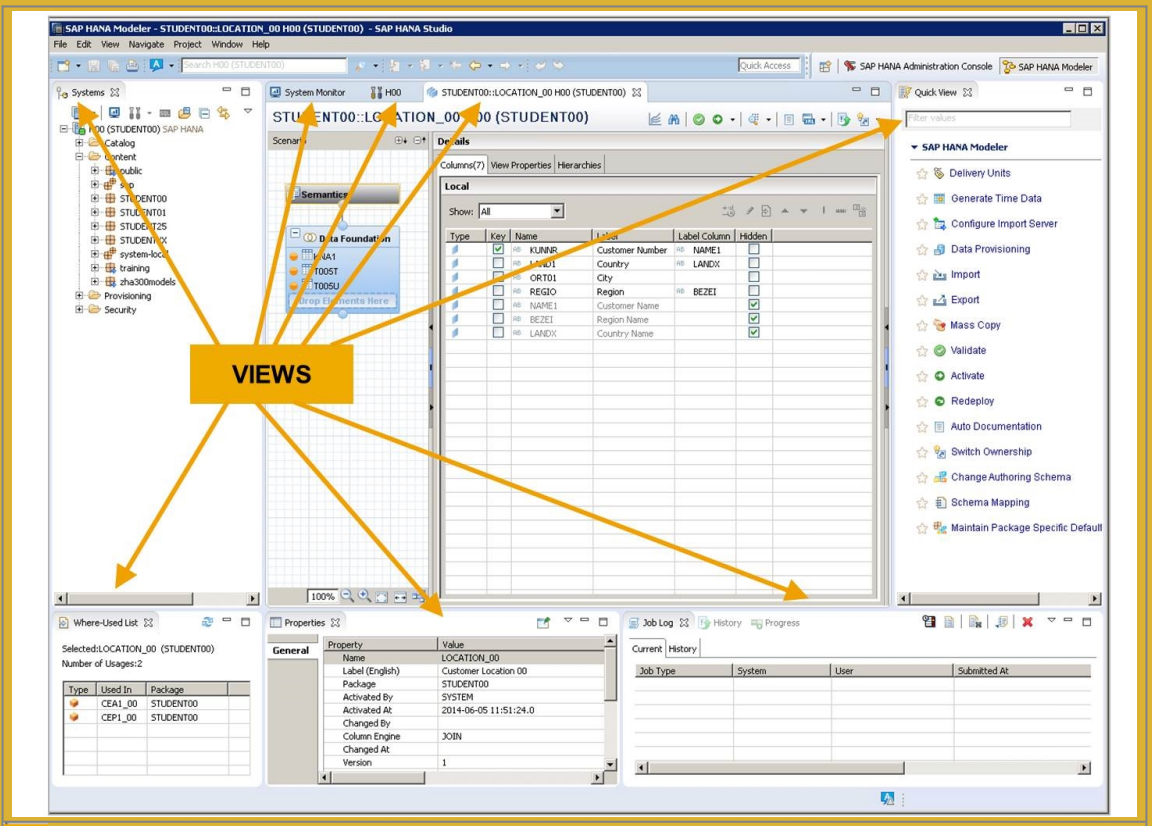


Figure 24: Perspectives Based on a Layout of Views

There are many other perspectives in the SAP HANA Studio.

To open a perspective, choose *Windows* → *Open Perspective*. Choose a perspective from the list or choose *Other...*.

It is possible to have several perspective open at the same time, and to switch from one perspective to another. To do so, in the top right of the screen, choose the perspective button you want to switch to. There, you can find a button that is a shortcut to the perspective selector.

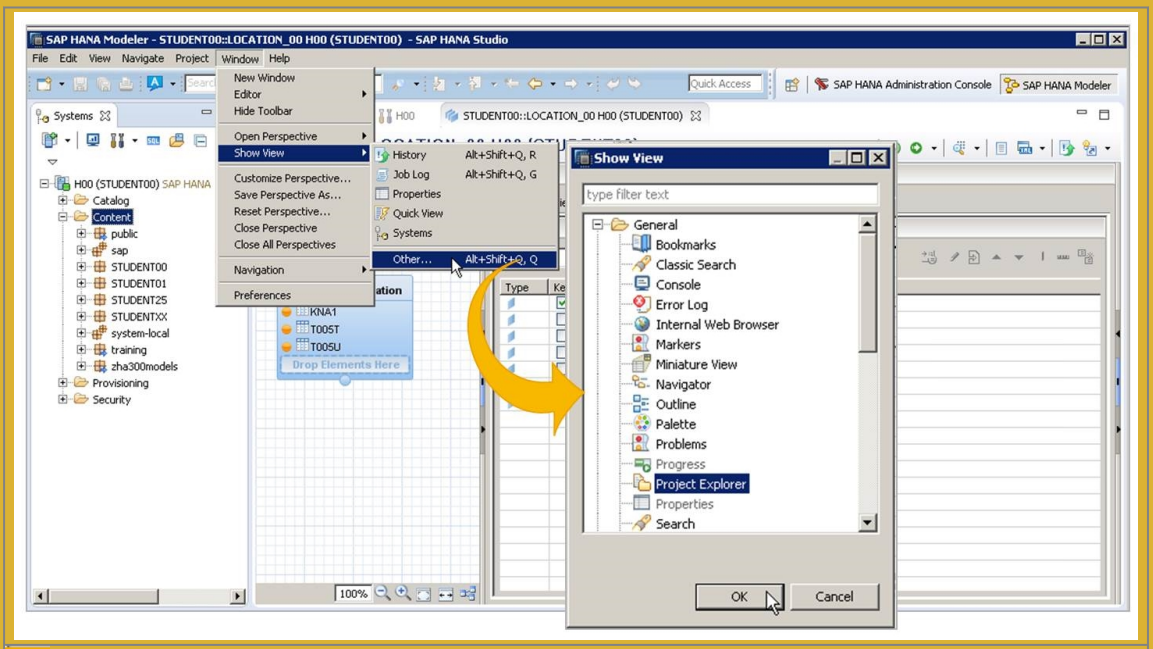


Figure 25: Adding a View to a Perspective

To add a view to a perspective, choose *Windows* → *Show View*. Choose a view from the list, or choose *Other...*. The view you have chosen is placed on your screen in a default location and with a default size but you can drag it to a new location as required and resize.

Any perspective can be reset to its default layout to restore the default views in their original positions and sizes. To do so, choose *Window* → *Reset Perspective...*.

The Systems View

Some views are more important than others. The *Systems* view is one of the most important so let's give it some focus here.

The *Systems* view is used to navigate the various content of SAP HANA. This includes the run–time contents of the SAP HANA database and also the design-time modeling content. You can also list the remote sources and their contents. You can list and manage users and roles.

In the *Systems* view, you find the connections. You can expand each connection and explore the contents. For each connection, the content is organized as follows:

- Catalog
- Content
- Provisioning
- Security

Catalog

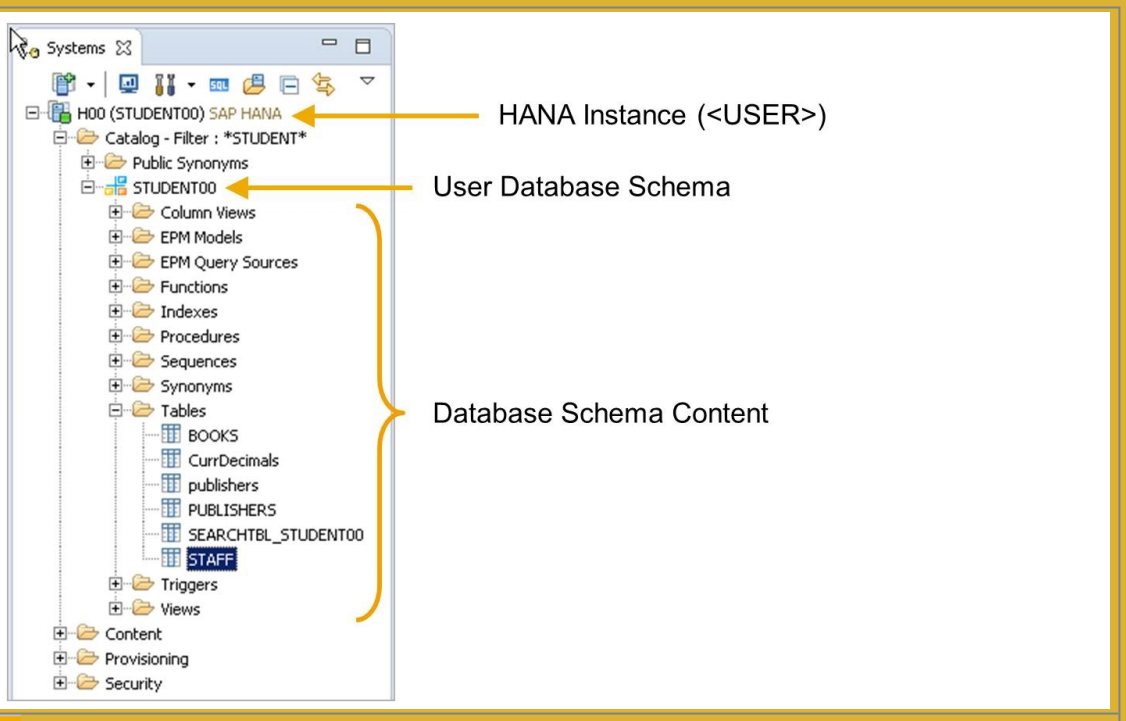


Figure 26: The Systems View: Catalog

The catalog contains all database objects such as tables, views, functions, and indexes. All these objects are organized into schemas. Schemas are used to categorize database content according to customer-defined groupings that have a particular meaning for applications. Schemas are also used to define access rights to the database objects that they contain.

Content

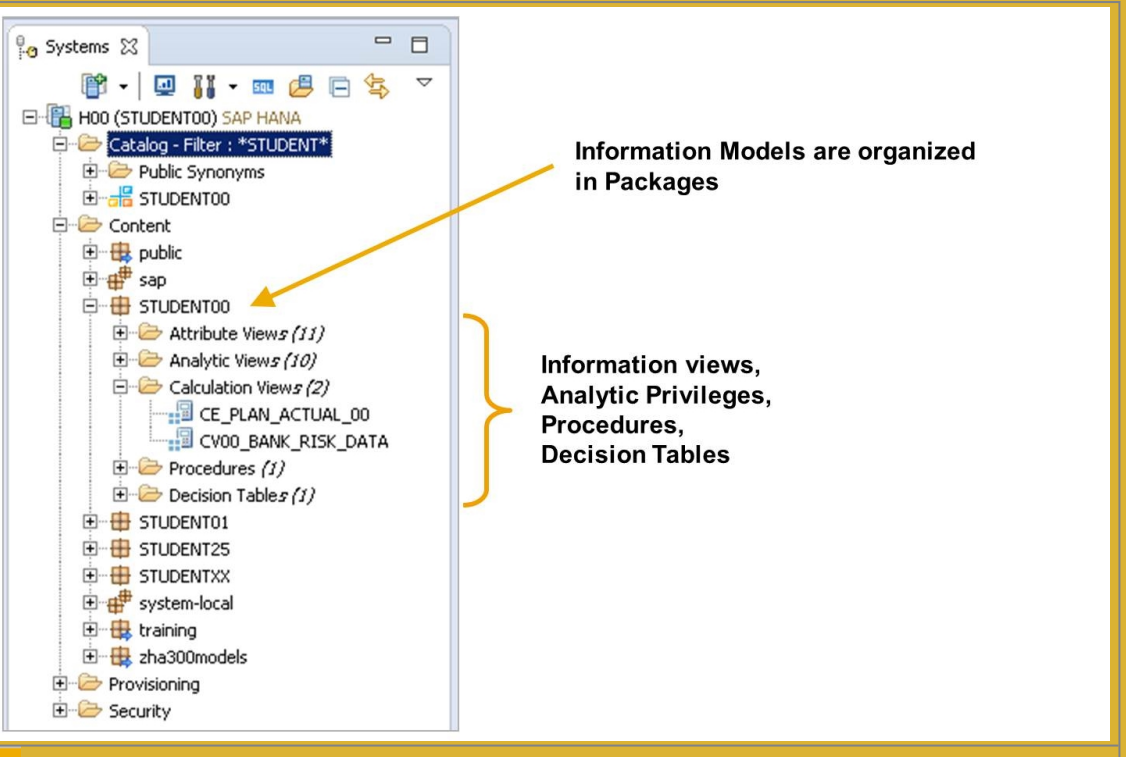


Figure 27: The Systems View: Modeling Content

The *Content* folder is where you store all the HANA-specific design-time data modeling objects such as calculation views and procedures. The modeling objects are organized in packages. Packages are used to define privileges to the developers to allows access only to the packages they are supposed to use. Packages are also used to transport the related objects to different SAP HANA systems.

Provisioning

The *Provisioning* folder is related to Smart Data Access (SDA). SDA is a data provisioning approach in which you can combine data from remote sources (Hadoop, SAP ASE, SAP IQ) with data of your SAP HANA physical tables, by exposing them as virtual tables.

Security

In the *Security* folder, the System Administrators define users and roles.

What you will not find in the *Systems* view are application development artifacts such as Javascript files, HTML code, core data services, and graphics used in application GUIs (such as buttons, and so on.). Development artifacts are found under the *Project* view which is part of the *Development* perspective.

The Administration View

The SAP HANA Administration View is another important view so lets focus a little on this.

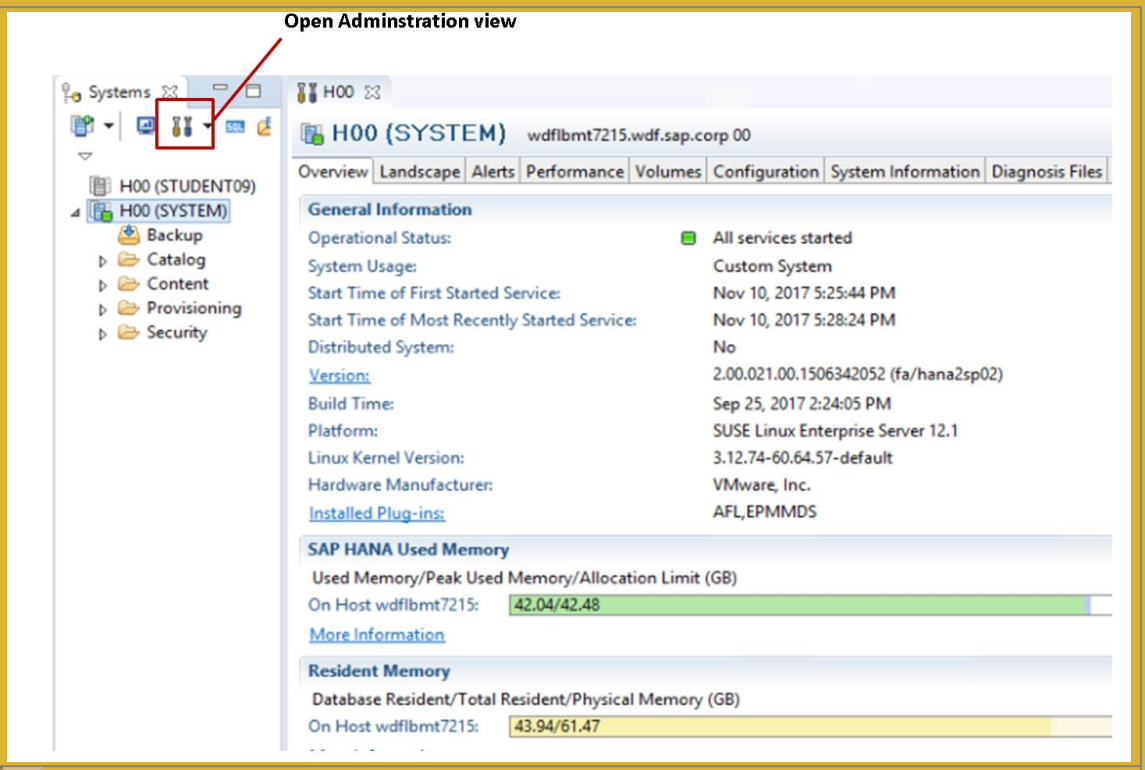


Figure 28: Administration View of HANA Studio

The *Administration* view is part of the *Administration* perspective. This is the main view used to administrate SAP HANA systems. In this view, you can do the following tasks:

- Start and stop a system
- Configure a system
- Monitor a system

- Backup and restore a system
- Perform a problem analysis

To open the *Administration* view, you can either double-click the required SAP HANA system listed in the *System Monitor* view or use the button in the toolbar of the *Systems* view.

Quick View

The *Quick View* contains a number of popular functions and basically puts them into a mini–menu for your convenience. When you want to perform an action from the *Quick View*, you must first select the SAP HANA system on which the action is executed.

To access the *Quick View* you can:

- Choose *Help* → *Quick View*.
- Reset the *Modeler* perspective.

Note:
The *Quick View* only displays within the *Modeler* perspective.

In summary, with the different perspectives and views of SAP HANA Studio, you can easily reach all the tools to develop complete applications from one interface. This ranges from the creation of database tables, to developing models to expose the table data in meaningful views, to developing applications that consume the views, and finally to monitoring performance of the applications.

Get Started with SAP HANA Studio

51

The purpose of this exercise is to set up and become familiar with the SAP HANA Studio user interface. You will perform the following tasks:

- Start SAP HANA Studio.
- Add an SAP HANA connection to the *Systems* view.
- Define SAP HANA Studio preferences.
- Work with *Views* and *Perspectives*.

Business Example

You are working on a customer project where SAP HANA is installed. You need to become familiar with SAP HANA Studio and learn how to connect to the SAP HANA system and browse the database catalog objects as well as the repository.

1. Launch the SAP HANA Studio.

   If you are prompted to choose a folder to store settings, use the default location and choose *Submit*.

   If you are prompted to choose a workspace folder, leave the defaults unchanged and choose *OK*.

   If you are asked to create a password hint (in case you forget your password), choose *No*.

2. From the *Welcome* page, open the *SAP HANA Administration Console* perspective.

3. Connect your SAP HANA Studio to the SAP HANA training system using the details shown in the table, *Connection Information*.

   Table 2: Connection Information

Field	Value
Host Name	**`wdflbmt7215.wdf.sap.corp`**
Instance Number	**`00`**
Database Mode	*Multiple containers*
Database name	**`H00`** (Tenant Database)
Description	**`My SAP HANA Training system`**

4. Enter the credentials shown in the table, *User Credentials*.

Table 3: User Credentials

Field	Value
User	*STUDENT##* (don't forget to swap ## for your 2 digit number)
Password	**`Training1`**

5. Set your SAP HANA Studio preferences using the values shown in the table, *Preferences*.

Table 4: Preferences

Path and Parameters	Value
General → Network Connections → Active Provider	**`Direct`**
General → Web Browser → Use External Web Browser	**`Internet Explorer`**

6. You are currently viewing the *SAP HANA Administrator* perspective which is ideal for operations people. Now switch to the *SAP HANA Modeler* perspective to see what the layout looks like for data modelers.

Note:
Right-click on any perspective button and choose *Show Text* so each perspective button displays its label.

7. Switch back to the *SAP HANA Administration Console* perspective for the next tasks.
8. Explore the *Systems* view by first increasing the width of the view to occupy half the screen and then expand each of the following nodes in turn.
   - *Catalog*
   - *Content*
   - *Provisioning*
   - *Security*

Note:
You can temporarily expand any view in the perspective to full-screen by simply double-clicking in the centre of the tab. You can restore the view to its original size by repeating this action. This can be very helpful as many of the views contain a lot of information that is much easier to read when you expand it to fill the entire screen.

9. Open the *Administration* view for your SAP HANA system.

Get Started with SAP HANA Studio

53

The purpose of this exercise is to set up and become familiar with the SAP HANA Studio user interface. You will perform the following tasks:

- Start SAP HANA Studio.
- Add an SAP HANA connection to the *Systems* view.
- Define SAP HANA Studio preferences.
- Work with *Views* and *Perspectives*.

Business Example

You are working on a customer project where SAP HANA is installed. You need to become familiar with SAP HANA Studio and learn how to connect to the SAP HANA system and browse the database catalog objects as well as the repository.

1. Launch the SAP HANA Studio.

   If you are prompted to choose a folder to store settings, use the default location and choose *Submit*.

   If you are prompted to choose a workspace folder, leave the defaults unchanged and choose *OK*.

   If you are asked to create a password hint (in case you forget your password), choose *No*.

   a) Locate and launch SAP HANA Studio with the default settings.

2. From the *Welcome* page, open the *SAP HANA Administration Console* perspective.

   a) Choose *Open SAP HANA Administration Console*.

3. Connect your SAP HANA Studio to the SAP HANA training system using the details shown in the table, *Connection Information*.

   Table 2: Connection Information

Field	Value
Host Name	**`wdflbmt7215.wdf.sap.corp`**
Instance Number	**`00`**
Database Mode	*Multiple containers*
Database name	**`H00`** (Tenant Database)
Description	**`My SAP HANA Training system`**

   a) In the *Systems* view, right-click the blank area and choose *Add System*.

b) Enter the details from the table, Connection Information, in the appropriate fields.

c) Choose *Next*.

4. Enter the credentials shown in the table, *User Credentials*.

Table 3: User Credentials

Field	Value
User	*STUDENT##* (don't forget to swap ## for your 2 digit number)
Password	**`Training1`**

a) Enter your user name and password shown in the table, *User Credentials*.

b) Choose *Finish*.

5. Set your SAP HANA Studio preferences using the values shown in the table, *Preferences*.

Table 4: Preferences

Path and Parameters	Value
General → Network Connections → Active Provider	**`Direct`**
General → Web Browser → Use External Web Browser	**`Internet Explorer`**

a) Choose *Window → Preferences* and follow the path provided in the table, *Preferences*, to set the values.

b) When finished, select *OK*.

6. You are currently viewing the *SAP HANA Administrator* perspective which is ideal for operations people. Now switch to the *SAP HANA Modeler* perspective to see what the layout looks like for data modelers.

a) To display all available perspectives, choose *Window → Perspective → Open Perspective → Other*.

b) Choose the *SAP HANA Modeler* perspective.

Note that the layout changes slightly to provide features that a modeler uses.

Note:
Right-click on any perspective button and choose *Show Text* so each perspective button displays its label.

7. Switch back to the *SAP HANA Administration Console* perspective for the next tasks.

a) Using the *perspective switch* buttons located in the upper-right corner of the screen, choose *SAP HANA Administration Console*.

8. Explore the *Systems* view by first increasing the width of the view to occupy half the screen and then expand each of the following nodes in turn.

- *Catalog*
- *Content*
- *Provisioning*
- *Security*

Note:
You can temporarily expand any view in the perspective to full-screen by simply double-clicking in the centre of the tab. You can restore the view to its original size by repeating this action. This can be very helpful as many of the views contain a lot of information that is much easier to read when you expand it to fill the entire screen.

a) Drag the right vertical border of the *Systems* view towards the center of the screen to increase its size.

b) In the *Systems* view, click the arrow to the left of each of the four folders to expand the details and explore the tree structure.

9. Open the *Administration* view for your SAP HANA system.

a) In the *Systems* view, highlight the *H00 (STUDENT##)* system.

b) Choose the Administration icon (looks like two screwdrivers) in the tool bar at the top of the *Systems* view.

c) Explore the system information for the highlighted system.

Web-based Development Workbench

SAP Web based Development Workbench

In addition to SAP HANA Studio, for XS developments there is the SAP HANA Web-based Development Workbench.

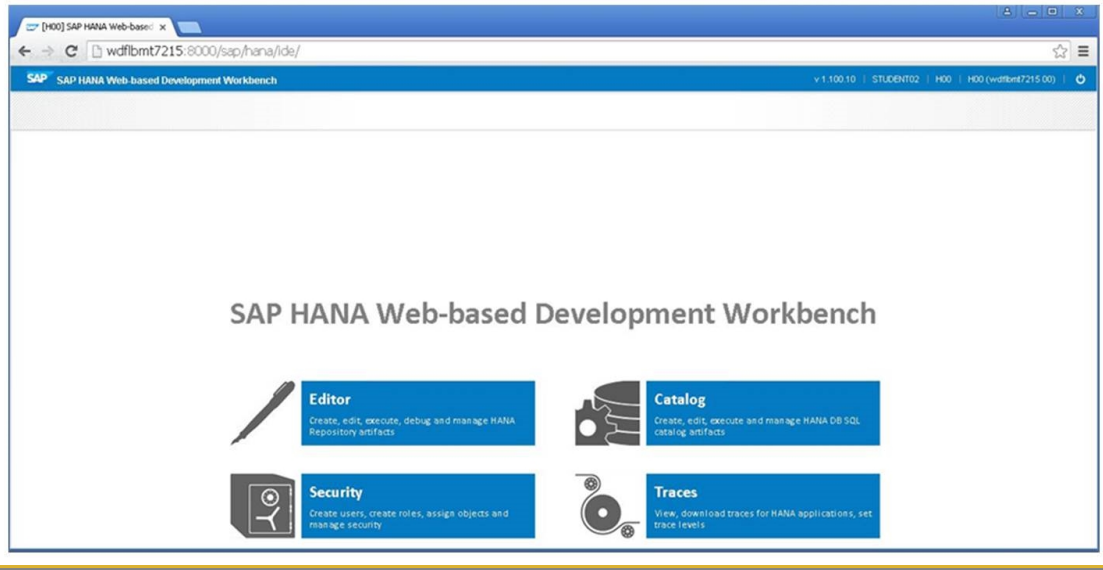


Figure 29: SAP Web Based Development Workbench

This lightweight web-based interface was aimed at developers and coders as an alternative to using the heavier SAP HANA Studio. The SAP HANA Web Development Workbench focused only on the features needed by application developers. You cannot use this interface with XSA developments.

> Note:
> Be careful not to confuse SAP Web based Development Workbench with the Web IDE for SAP HANA.

In order to get started with the SAP Web based Development Workbench, open a web browser and enter the following URL:

`http://<hostname>:80<instance>/sap/hana/xs/ide`

Substitute the two placeholders with your system details.

Enter your SAP HANA user and password and choose *LOG ON*.

If there is interest, you could show the Web Based Development Workbench in our training landscape. The URL for our landscape is http://wdflbmt7215:8000/sap/hana/xs/ide

SAP Web IDE for SAP HANA

SAP Web IDE for SAP HANA is a browser-based, integrated development environment (IDE). It is used for the development of SAP HANA-based applications that are comprised of web-based or mobile user interfaces (for example, HTML), business logic (for example, JAVA), and extensive SAP HANA data models (for example, calculation views). SAP Web IDE for SAP HANA works in conjunction with the SAP HANA deployment infrastructure (HDI) and the XS

Advanced runtime platform. It does not work with XS Classic development and so it is associated with SAP HANA 2.0 developments.

Note:
It is important to remember that SAP Web IDE for SAP HANA is not the same as SAP Web IDE. SAP Web IDE is a similar client tool, but it is used for the development of SAP Fiori-based applications and SAPUI5 applications. SAP Web IDE for SAP HANA is used to build XSA applications on SAP HANA. Be careful when referring to documentation and online references (especially from unofficial SAP sources) as these products are easily confused.

SAP Web IDE for SAP HANA provided the following features:

- **Integrated Workspace and Project Management**
  - Full workspace management on the server
  - Integrated Git-based version control
  - A dedicated template and wizards for multi-target application projects
- **Development Tools**

  For SAP HANA database artifact development:
  - Graphical modelers for complex artifacts, such as data models or calculation views
  - An SQL console and an MDX console in the database explorer
  - Integrated browsing of the database catalog in the database explorer
  - Syntax highlighting and content assistance for selected artifacts
  - Integrated performance analysis tools for SQL script and calculation views
- **Build, Run, and Deploy**
  - Incremental build support
  - Structured console output and logging
  - Automatic creation of development sandboxes on SAP HANA (HDI containers)
  - Automatic provisioning of HDI container services
  - Generation of deployment archives

To launch the Web IDE for SAP HANA, you need to obtain the URL from the administrator. This includes the host name where HANA is installed, which is unique for each installation. After launching the URL, the screen shown in the figure, *Launching Web IDE for SAP HANA*, displays.

Launching Web IDE for SAP HANA

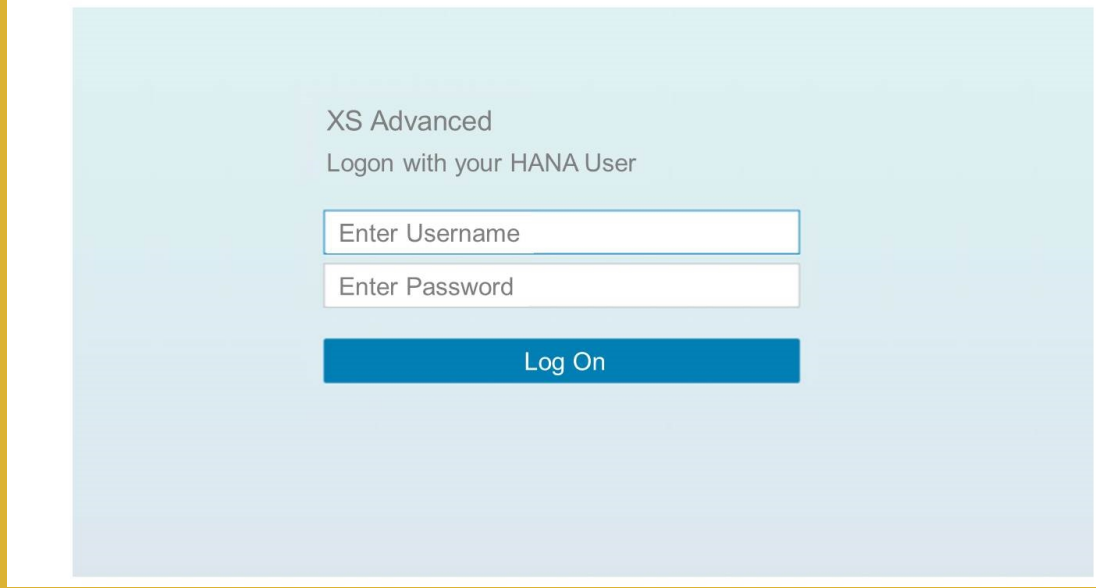


Figure 30: Launching Web IDE for SAP HANA

Once you have logged on, you can then decide which view you want to work in.

Switching Views in Web IDE

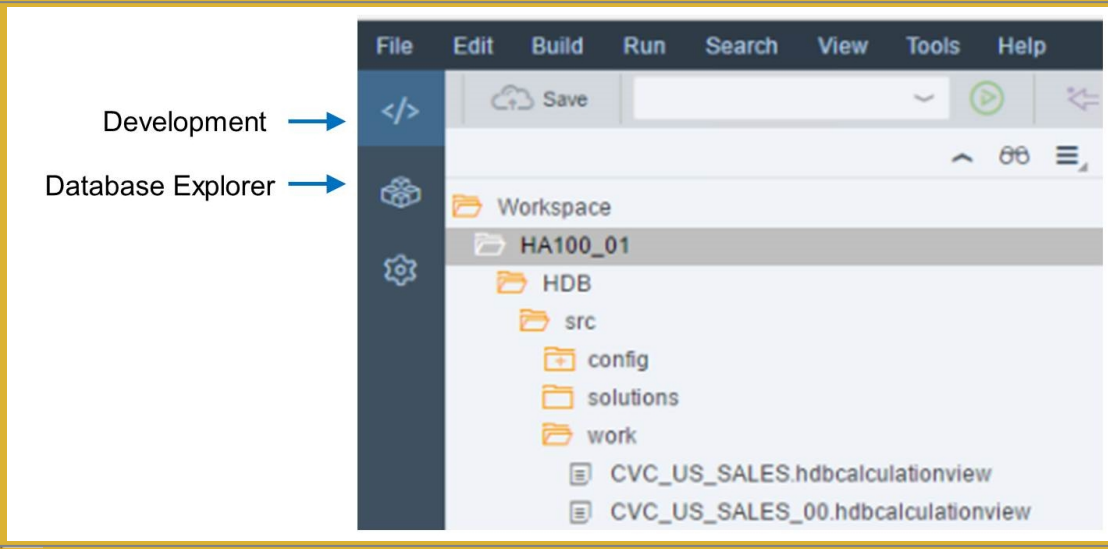


Figure 31: Switching Views in Web IDE

There are two main views in Web IDE for SAP HANA. These are the *Development* view and the *Database Explorer* view.

Development View of Web IDE for SAP HANA

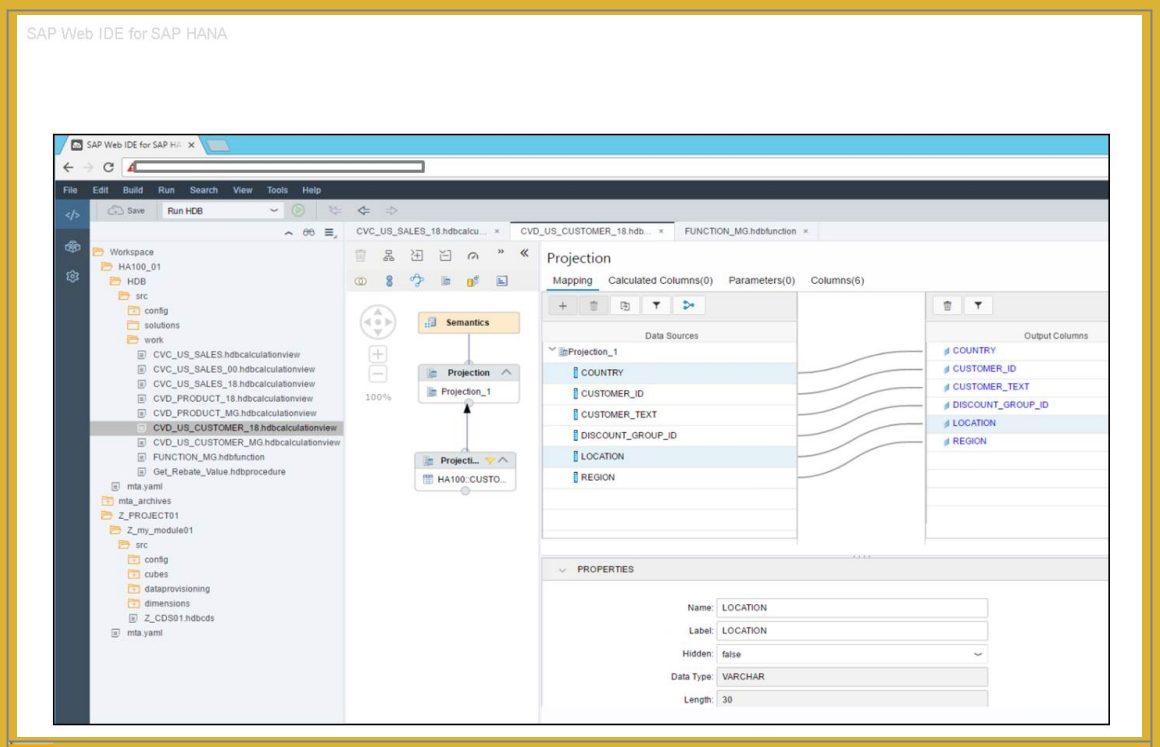


Figure 32: Development View of Web IDE for SAP HANA

The *Development* view is used by application developers to write code. It is also used by modelers to develop data models. In comparison to SAP HANA Studio, it is like having the *Modeling* perspective and the *Development* perspective combined.

The *Development* view has some convenient built-in features for developers, such as code completion, debuggers and syntax error highlighting.

Creating a New Development Module

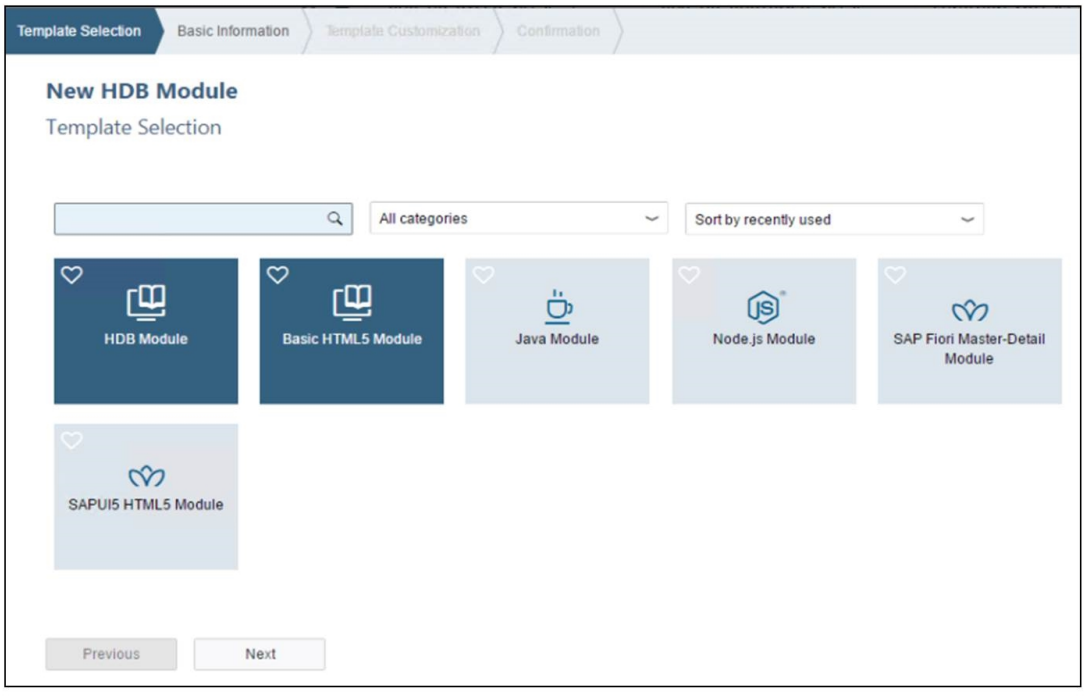


Figure 33: Creating a New Development Module

To get started with modeling or application development, you first need to create a development module. This provides the templates and structure that you need.

Once your module is built, you can add additional folders to provide better organization of your content.

To create the various development artifacts and source files, you right-click a folder and choose the type of file you want to create from the context menu.

As an example, if you were a modeler you would be able to create the following:

- Procedure
- Function
- Calculation View
- Flowgraph
- Analytic Privilege
- CDS Artifact
- Text Analysis dictionaries and rule sets

Web IDE development is completely file based, which means that each source artifact is a simple file (identified with an extension) that is easy to export and import within and also across SAP HANA.

Each file you create in Web IDE has its own extension, such as ".hdbfunction" or ".hdbcalculationview", which makes it easy to identify its type. This is also how Web IDE knows which editor to open for each type of artifact.

Exploring the Database with Web IDE

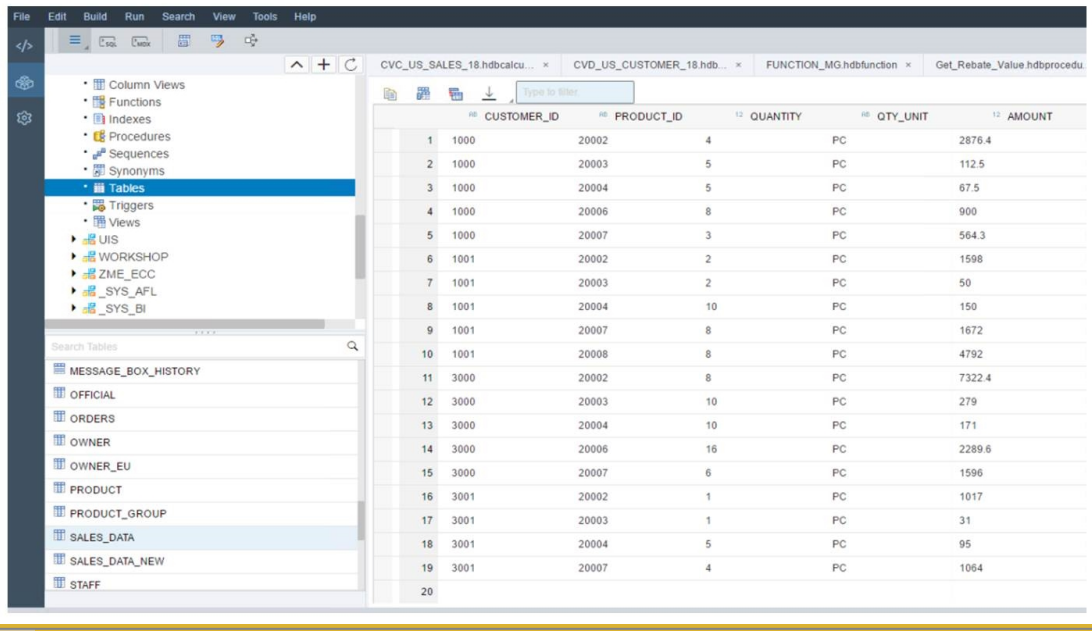


Figure 34: Exploring the Database with Web IDE

The *Database Explorer* view is used by anyone who needs to view the runtime database objects such as tables, views, and functions. There are tools to view the metadata and also the data of tables.

The *Database Explorer* also includes an SQL and MDX console so you can enter and execute statements directly against the database.

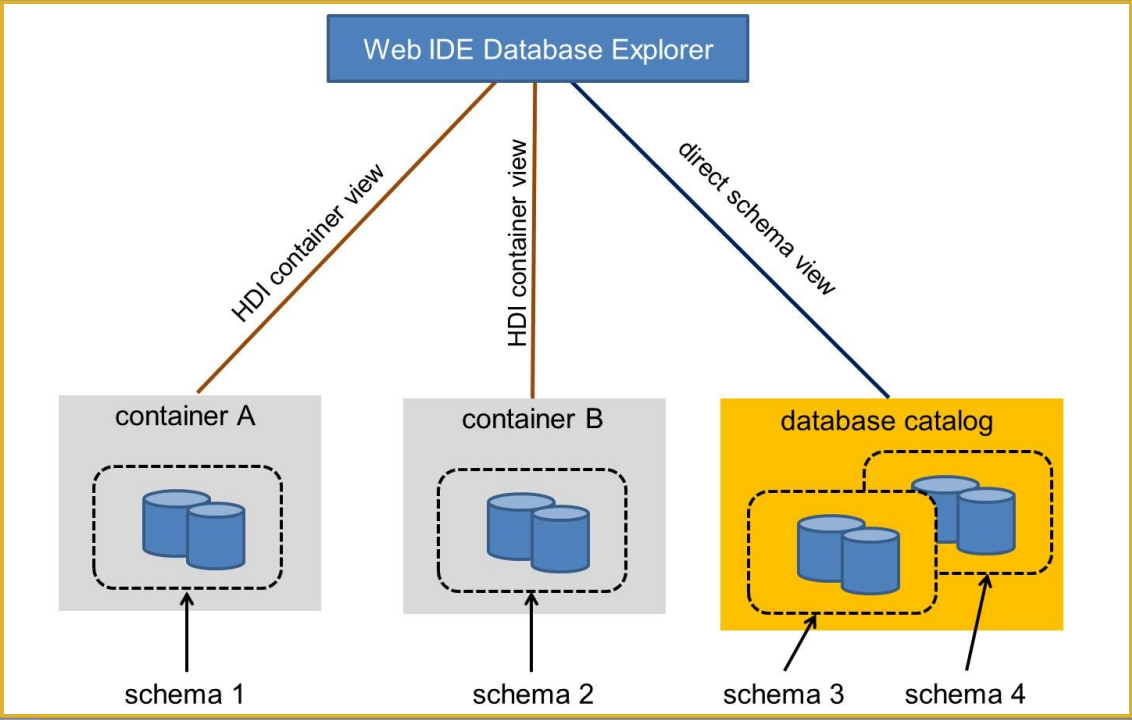


Figure 35: The HDI Container or Schema View

There are two ways to view the database in the Database Explorer; either using the HDI container infrastructure or the classic schema-based catalog. You can switch between the two views. To get started with the *Database Explorer*, you need to first add a database entry to the navigation pane on the top-left side of the screen. Whilst doing this you are asked if it is based on the classic schema or new HDI container.

After adding the database entry, you can then browse the database objects.

The classic schema based approach is the traditional way we view the list of database schemas and then expand these to reveal the database objects they contain. This is the only way you can view the database using SAP HANA Studio.

But let's briefly describe a **HDI container** as this is the newer approach to accessing the database, and whilst it may seem a little complex at first, you soon discover that this approach is simple and makes much better sense for flexible development and deployment.

A HDI container (or sometimes called a schema-container or just container) is a logical 'box' that sits above the physical database schema. The basic idea is to allow development to be de-coupled from the physical database storage components. Or, put simply, the developers never need to be concerned about the physical database deployment. Actually, when a container is first built, a physical database schema is automatically generated and bound this container. So developers simply refer to the container and they must never specify the physical schema name. (This is called schema-less development).

The HDI container is a key component in the new HANA deployment infrastructure (HDI). HDI is only used with XSA, not XS.

When created for the first time, a container generates its own physical schema with a long technical name. You can create database objects directly in your container and they are physically placed in a generated physical schema.

One of the key benefits of getting away from tying development to physical schemas and instead using containers, is that different versions of the database can co-exist in the same SAP HANA system. For example, you could upgrade an application (but keep the older version running), and redeploy the new version along with its improved database objects and a different schema is used for the newer applications database objects. The older application continues to run alongside using its own database schema. Containers isolate deployments of applications in the same SAP HANA platform.

Note:
With XS development, all database objects from all applications were deployed to one giant fixed system schema (called _SYS_BIC). You could not choose your own deployment schema. This meant you could never have multiple versions of the database objects supporting different versions of applications as all updates would overwrite the previous objects as they all lived in the same schema.

Unit 2
Exercise 3

Get Started with Web IDE for SAP HANA

63

The purpose of this exercise is to launch, and become familiar with SAP Web IDE for SAP HANA. Perform the following tasks:

- Launch Web IDE for SAP HANA.
- Navigate between the *Development* and *Database Explorer* views.
- Import a project to your workspace.
- Add a database to the Database Explorer.

The reason that we are importing a ready-made XSA project instead of creating our own from scratch, is because we want to provide some ready-made calculation views that we can open quickly to show some navigation. If each student had created their own project (using 'New') then the project would be empty until students create their first objects. Also, there are a number of essential settings that need to be defined when creating a brand new project (such as defining namespaces, building synonyms etc.) that are too detailed to explain in this overview course. Our imported project already contains all setting that are required so we don't have to get into explanations here (those things are covered in detail in HA300). Remember, when using Web IDE, we are working with XSA projects which, unlike XS projects, use a different type of design time framework, and you simply cannot just navigate to existing objects (tables, functions, calculation views) in the database as we did in Studio. In Web IDE, each person works in their own isolated workspace in a project and each project needs carefully setting up. So it might be helpful to explain to the students why we are fast-tracking them by importing a ready-made project, in case they assume they always need to do an import to get started on any project.

Business Example

You are joining a customer project where SAP HANA XSA developments are in progress, so you need to become familiar with SAP Web IDE for SAP HANA.

Task 1: Log on to Web IDE for SAP HANA

1. Launch the Web IDE for SAP HANA.
2. Log on using the details shown in the table, *Logon Details*. Allow Google Chrome to save your password so you don't have to re-enter this next time you need to logon.

Table 5: Logon Details

Field	Value
User	**`STUDENT##`**
Password	**`Training1`**

Task 2: Import a ready-made XSA project and build the runtime database objects

Setting up a brand new Web IDE project requires a number of configuration steps. To keep things simple in this class, and to get started quickly, import a ready-made project that already has many of the settings in place. It also includes all the exercise solutions for this class.

1. Import an existing XSA project into your workspace.

   The project archive, *HA100.zip*, is located in your *HA100* folder, and must be imported to your workspace with the project name **`HA100_##`** (where ## is your group number).

2. Assign your project to the *DEV* space. This is the only setting that cannot be pre-defined in an imported project and you have to do this manually.

3. Build the *HDB* module of your project so that the database container can be generated.

Caution:
Make sure you execute the build at the HDB module level. If you build the project, the HDB module will not be built.

4. Expand the *src* folder in your imported project until you reach the *solutions* sub–folder, where various development artifacts already exist.

5. Open the file, *CVC_US_SALES_00.hdbcalculationview*, in the graphical *calculation view editor*, and study the definition of the calculation view.

Note:
You can temporarily expand any view to full screen by simply double-clicking in the centre of the tab. You can restore the view to its original size by repeating this action. This can be very helpful as many of the views contain a lot of information that is easier to read when you fill the whole screen.

6. Close the calculation view *CVC_US_SALES_00.hdbcalculationview*

Get Started with Web IDE for SAP HANA

65

The purpose of this exercise is to launch, and become familiar with SAP Web IDE for SAP HANA. Perform the following tasks:

- Launch Web IDE for SAP HANA.
- Navigate between the *Development* and *Database Explorer* views.
- Import a project to your workspace.
- Add a database to the Database Explorer.

The reason that we are importing a ready-made XSA project instead of creating our own from scratch, is because we want to provide some ready-made calculation views that we can open quickly to show some navigation. If each student had created their own project (using 'New') then the project would be empty until students create their first objects. Also, there are a number of essential settings that need to be defined when creating a brand new project (such as defining namespaces, building synonyms etc.) that are too detailed to explain in this overview course. Our imported project already contains all setting that are required so we don't have to get into explanations here (those things are covered in detail in HA300). Remember, when using Web IDE, we are working with XSA projects which, unlike XS projects, use a different type of design time framework, and you simply cannot just navigate to existing objects (tables, functions, calculation views) in the database as we did in Studio. In Web IDE, each person works in their own isolated workspace in a project and each project needs carefully setting up. So it might be helpful to explain to the students why we are fast-tracking them by importing a ready-made project, in case they assume they always need to do an import to get started on any project.

Business Example

You are joining a customer project where SAP HANA XSA developments are in progress, so you need to become familiar with SAP Web IDE for SAP HANA.

Task 1: Log on to Web IDE for SAP HANA

1. Launch the Web IDE for SAP HANA.
   a) To launch the Web IDE logon, double-click the shortcut *Web IDE for SAP HANA* in the folder *Favorites* → *HA100* → *URLs*.
2. Log on using the details shown in the table, *Logon Details*. Allow Google Chrome to save your password so you don't have to re-enter this next time you need to logon.

Table 5: Logon Details

Field	Value
User	**STUDENT##**

Field	Value
Password	**`Training1`**

a) At the logon screen, enter the details shown in the table, *Logon Details*, and choose *Log On*.

b) In the Google Chrome dialog box that prompts you to save your password, choose *Save*.

c) In the *Tips and Tricks* dialog box choose *Close*.

Task 2: Import a ready-made XSA project and build the runtime database objects

Setting up a brand new Web IDE project requires a number of configuration steps. To keep things simple in this class, and to get started quickly, import a ready-made project that already has many of the settings in place. It also includes all the exercise solutions for this class.

1. Import an existing XSA project into your workspace.

   The project archive, *HA100.zip*, is located in your *HA100* folder, and must be imported to your workspace with the project name **`HA100_##`** (where ## is your group number).

   a) Right click the *Workspace* folder, which is found under the *Development* view, and choose *Import* → *File or Project*.

   b) Choose *Browse* and navigate to the Windows folder *Favorites* → *HA100*.

   c) Select the file *HA100.zip* and choose *Open*.

   d) In the *Import to* field, enter **`/HA100_##`** (where ## is your group number).

   e) Leave the *Extract Archive* checkbox checked to ensure that the zip file is unzipped during import.

   f) Choose *OK* to begin the import.

2. Assign your project to the *DEV* space. This is the only setting that cannot be pre-defined in an imported project and you have to do this manually.

   a) In the workspace, right-click your *HA100_##* project and choose *Project Settings*.

   b) Open the *Space* panel and choose *DEV* from the drop-down list.

   c) Choose *Save*.

   d) Choose *Close*.

3. Build the *HDB* module of your project so that the database container can be generated.

> Caution:
> Make sure you execute the build at the HDB module level. If you build the project, the HDB module will not be built.

   a) Right-click the *HA100_##* → *HDB* folder and choose *Build*.

   b) Wait a few moments, then check that the log shows a successful build.

4. Expand the *src* folder in your imported project until you reach the *solutions* sub–folder, where various development artifacts already exist.
   a) Expand the folder path *HA100_##* → *HDB* → *src* → *solutions*.
5. Open the file, *CVC_US_SALES_00.hdbcalculationview*, in the graphical *calculation view editor*, and study the definition of the calculation view.
   a) In the *solutions* sub-folder, double-click the file *CVC_US_SALES_00.hdbcalculationview* and a new tab appears.

Note:
You can temporarily expand any view to full screen by simply double-clicking in the centre of the tab. You can restore the view to its original size by repeating this action. This can be very helpful as many of the views contain a lot of information that is easier to read when you fill the whole screen.

6. Close the calculation view *CVC_US_SALES_00.hdbcalculationview*
   a) Right-click on the tab and chose *Close*, or click the 'X' to the right of the tab.

SAP HANA Cockpit

SAP HANA cockpit provides a single point of access to a range of tools for administration and monitoring of multiple, individual, and tenant SAP HANA databases. It also integrates the SQL development capabilities required by administrators. SAP HANA cockpit is a web-based HTML5 user interface that you access through a browser. It runs on SAP HANA XSA.

You can use the cockpit to monitor and manage systems running SAP HANA 2.0 or SAP HANA 1.0 SPS 12.

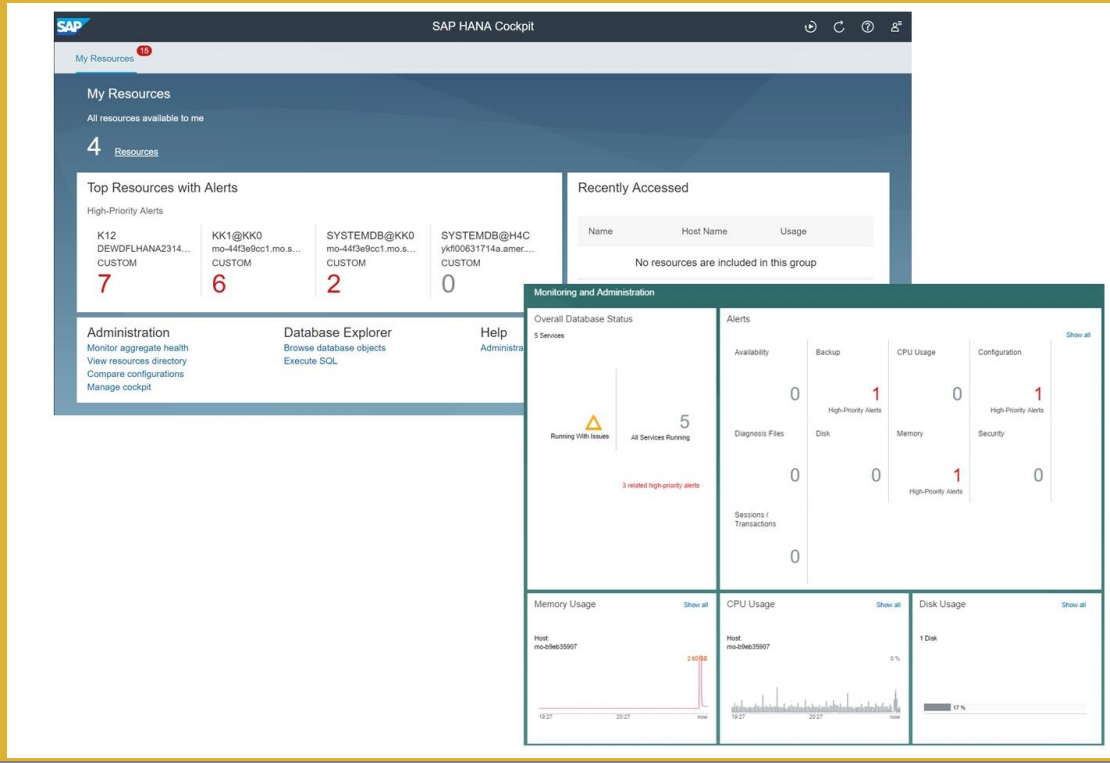


Figure 36: SAP HANA Cockpit 2.0

Advantages of SAP HANA Cockpit 2.0

The SAP HANA cockpit provides aggregate, system, and database administration features. Examples of these include database monitoring, user management, and data backup. Administrators can use the SAP HANA cockpit to start and stop services, to monitor the system, to configure system settings, and to manage users and authorizations. Cockpit pages that allow you to manage SAP HANA options and capabilities (for example, SAP HANA dynamic tiering) are only available if the option or capability has been installed.

Initially, the SAP HANA cockpit displays data at a landscape or enterprise level. You can quickly drill down to an overview of an individual resource. Links, data, tiles, and different parts of a single tile are drillable, which provides access to more detailed information and functions.

Integrated into the cockpit is the SAP HANA database explorer. The database explorer allows you to query information about the database. It does this using the in-built SQL console or MDX console, as well as the ability to view information about your database's catalog objects.

There are actually two releases of the SAP HANA Cockpit; 1.0 and 2.0. SAP HANA 2.0 Cockpit includes all the features of SAP HANA 1.0 Cockpit. In addition, SAP HANA 2.0 Cockpit allows you to manage and monitor multiple SAP HANA systems from a single location. You can manage both on-premise and cloud deployments of SAP HANA from the same cockpit.

SAP HANA Cockpit 2.0 Tasks

SAP HANA Cockpit 2.0 is a standalone component and it runs on its own HANA Express instance. It is not installed on the core SAP HANA platform. SAP HANA Cockpit 2.0 can be used to manage multiple SAP HANA instances, both on-premise and cloud. As a comparison, SAP HANA Cockpit 1.0 was an embedded XS application that was installed on the core SAP HANA instance. It could only be used with one SAP HANA instance (the one it was installed on).

Note:
Full administration of SAP HANA is currently not possible with SAP HANA cockpit. Certain tasks continue to require SAP HANA studio. It is planned that more administration features will be moved from Studio to Cockpit 2.0 in future releases, so eventually Studio will not be required at all for administration.

SAP HANA 2.0 Cockpit is a browser-based application. To use it, you need the latest version of Google Chrome, Mozilla Firefox, Microsoft Edge, or Apple Safari. Microsoft Internet Explorer 11 is also supported.

We recommend that for production purposes, you run the SAP HANA Cockpit 2.0 on its own physical server. However, technically it can run on the same server, side-by-side with SAP HANA 2.0, or SAP HANA 1.0 SPS12.

LESSON SUMMARY

You should now be able to:

- Work with SAP HANA interfaces

Unit 2
Lesson 3

70

Understanding Key Features of SAP HANA Database

LESSON OVERVIEW
This lesson will introduce you to the various aspects of the SAP HANA database.

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Understand key features of SAP HANA database

Row and Column Stores

In this lesson we turn our attention to the heart of SAP HANA; the in-memory database.

SAP HANA includes a fully-relational, ACID-compliant database.

> Note:
> ACID is an acronym that means the database can support Atomicity, Consistency, Isolation, and Durability (ACID). This is a requirement of a database that must prove that it is 100% reliable for mission-critical applications. The database must guarantee data accuracy and integrity even when there are lots of simultaneous updates across multiple tables.

Most traditional enterprise relational database tables are row-based, as this is regarded as the optimal design for a transactional system. However, SAP HANA not only handles transactions but also analytics. For analytics, column tables are optimal. SAP HANA's database supports both row tables and column tables. Both table storage types are needed in a platform that handles transactional and analytical applications in the same database. Developers can decide which type of table they need to support their applications; that is, row or column.

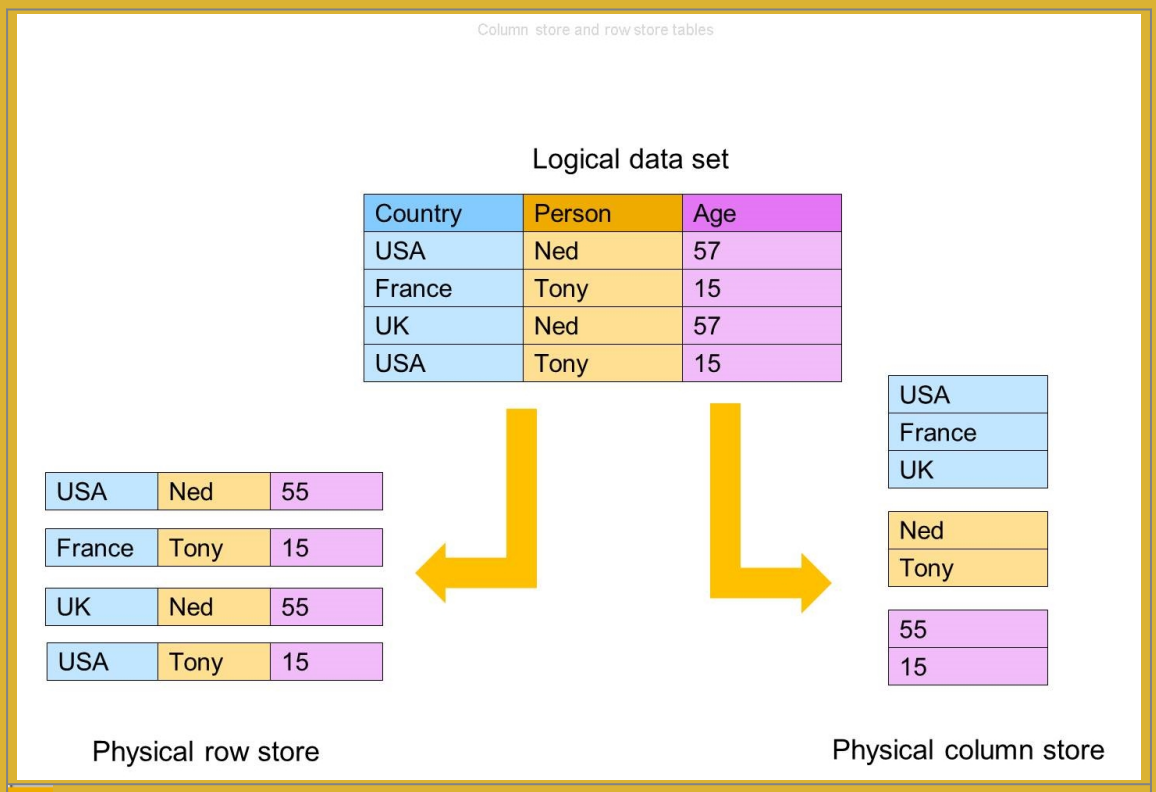


Figure 37: Column and Row Store

The figure, *Column and Row Store*, shows that the key difference between row and column stores is the way that the data is organized.

Column store tables are efficient for analytical applications where requests for sets of data are not predictable. Queries from analytical applications that are sent to the database often require only a subset of the overall data in the table. Usually only limited columns are required. With column store, only the required columns are loaded to memory, so you avoid using up memory with columns that will never be used. Also, the data is arranged efficiently with all values of a column appearing one after another. This continuous sequencing of the column values is preferred by the CPU, which is able to scan the values efficiently without having to skip over values.

With column store, SAP HANA scans columns of data so quickly that additional indexes are usually not required. This helps to reduce the complexity by avoiding the need to constantly create drop and rebuilding indexes.

It is easy to alter column store tables without dropping and reloading data.

Column store tables are optimal for parallel processing, as each core is able to work on a different column.

The downside to column store is the cost of reconstructing complete records from the individual column store, if all columns are required by the application. This is the case when the application is transaction-based and so, all fields are usually needed for a record update and must all be retrieved. This would be possible with column store. However, it would be slower than if the storage was row-based, where all the columns are always held together and can be read quickly.

Row storage is still needed to support transaction processing where all columns need to be retrieved. Often an application is both transactional and analytical. In this case, you must decide which is the best storage method to use. You cannot have a table that is both row and

column storage. It is easy to convert a table from row to column and the other way around, and you do not lose the data when doing this.

SAP S/4HANA combines transactional and analytical applications and so, utilizes both column and row store tables.

Data Footprint Reduction

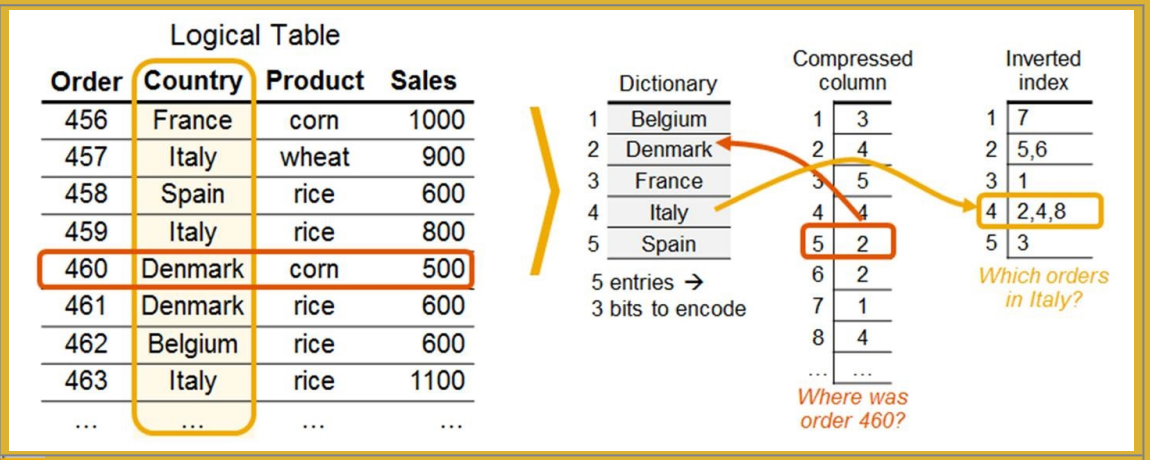


Figure 38: Compression

The data in the SAP HANA column store tables is automatically compressed. This is done to reduce the data footprint.

The following are some of the benefits of a reduced data footprint:

- You can get more data into CPU cache and therefore reduce main memory access, to maintain high performance.
- You can fit entire enterprise databases into memory and avoid disk access.
- Operations, such as backup and restore, are sped up as you deal with a smaller data size.

Mechanism of Compression

The amount by which data reduction can take place is driven by the shape of the business data. Compression is most impressive when there is a lot of repetition in the data values.

Take for example, a huge sales order table where the customer type A, B, or C is stored on each customer order. In this case, the customer type would appear a huge number of times in the column.

Compression strips out the repetition and stores each unique value once in a dictionary store. SAP HANA then uses integers to represent the business values in the original store, as this takes up far less space and is very efficient for scanning. SAP HANA links the dictionary entries to the actual table using special reference stores. These reference stores identify the position where the original value was and its corresponding business value from the dictionary store. This mechanism is embedded deep in the SAP HANA database. The processing happens invisibly.

Parallel Processing of Data

One of the key enablers of SAP HANA's incredible performance is parallel processing. With recent hardware developments, especially new multi-core processors, we can build instant-response applications by spreading the processing tasks across all the cores.

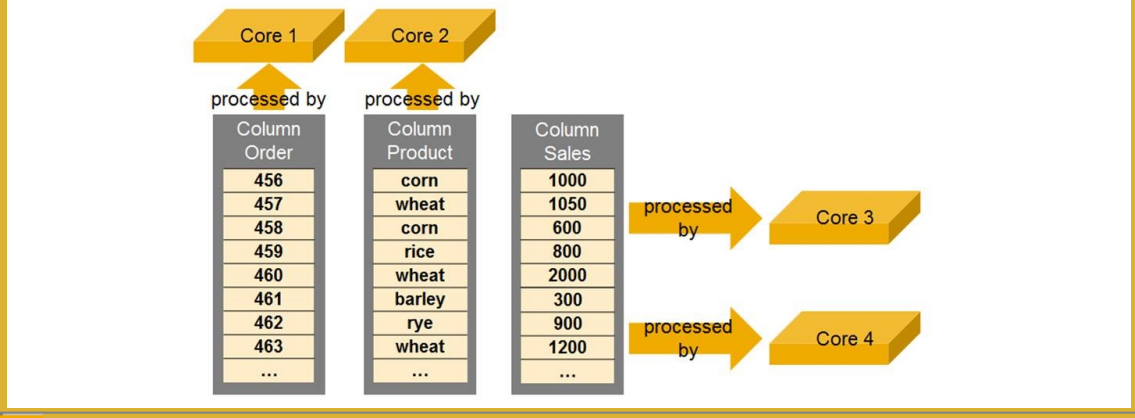


Figure 39: Parallel Processing

SAP HANA automatically spreads the workload across all cores and ensures that all parts of the hardware are contributing to the throughput.

SAP HANA is scalable. This means that you can easily add more processors as required, to increase the parallelization and therefore the speed of processing.

To take advantage of the built-in parallel processing capabilities of SAP HANA, you can use column store tables. Column store tables are automatically processed in parallel. Each column can be processed by one core. The more cores you add to the SAP HANA landscape, the more parallelization occurs.

For column store tables, you can also define partitions on each column. This means that only the required partitions are read to memory. For example, if a query requested only current-year data, then all other years in the column would be ignored. Partitions can be created based on known popular business values or by simply allowing SAP HANA to split large columns in an arbitrary way.

Note:
Up to and including SAP HANA 1.0 SPS09, the maximum number of partitions was 1,000 per column table. Since SAP HANA 1.0 SPS10 this has increased dramatically to 16,000 partitions per column table.

Parallel processing is a key enabler for real-time processing for any application powered by SAP HANA. By using column store tables with well-designed partitioning and data models, you can expect excellent performance from the database.

Data Temperatures

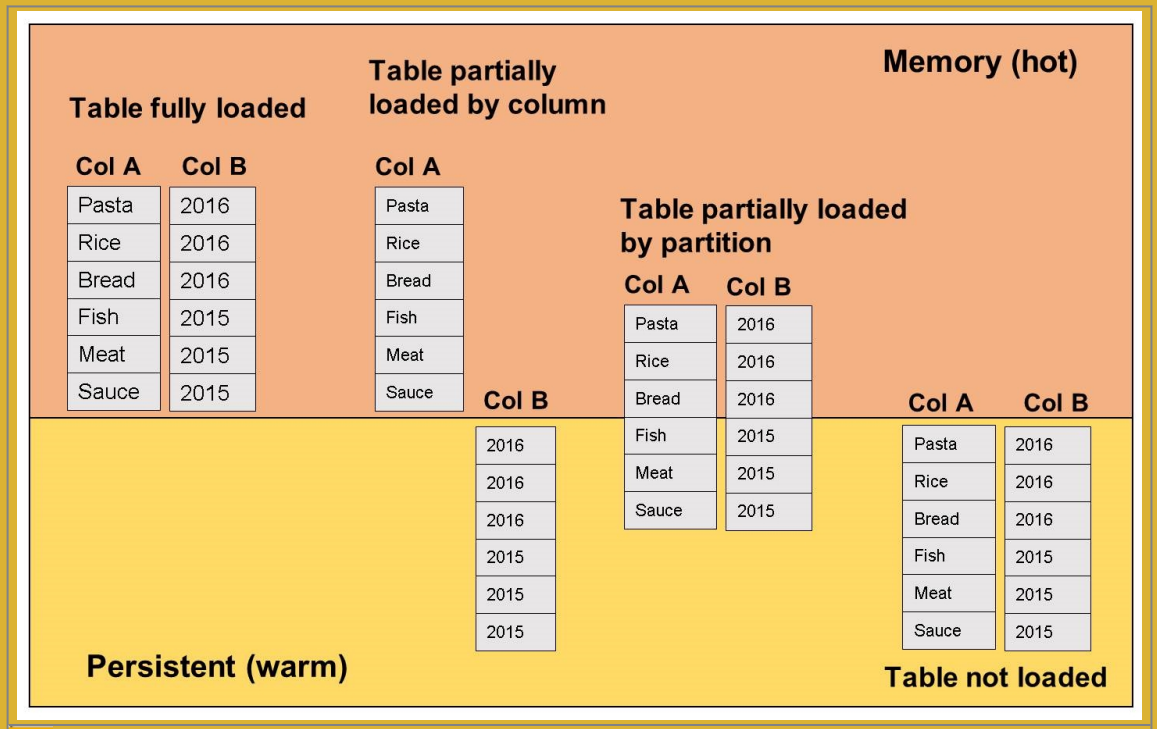


Figure 40: Data Distribution

SAP HANA is often described as an *in-memory database*. Although this is true, you must also remember that SAP HANA has a persistent storage layer too. The persistent storage layer is disk-based and is a mandatory part of SAP HANA. It is not a separate component.

There are two reasons why you need the disk layer. They are as follows:

- To provide an area to unload less important data when memory is full. We call the unloaded data *inactive data*.
- To enable data recovery if the power fails.

Note:
For now we are focusing on the first reason. We will cover the second reason later, when we discuss high availability.

Data Distribution

It is possible to fit an entire organization's data completely in memory if the SAP HANA memory is sized large enough. However, most organizations size their SAP HANA system with only enough memory to hold the most useful data, and they utilize disk to store the remaining data. This means that there is not enough memory for the entire organization's data and there will come a time when there may be competition for the memory. When memory is full, the data that is used less often is automatically moved from memory to disk to make way for new data. The larger the memory, the less displacement is needed. Remember also that some space is needed in memory as a working space for calculations.

Even though the cost of memory is falling and organizations can implement huge memory, it is still too expensive to be used to store all data of all ages. Data usually has a lifecycle. An

organization usually values their most recent data much more highly than their older data. Generally, organizations also access their recent data much more frequently than older data.

Conceptually, data can be classified into temperatures. Data that is accessed frequently is known as hot data. Data that is accessed less frequently is known as warm data. Data that is rarely accessed (often retained only for legal purposes) is known as cold data.

Data Temperature Storage

SAP HANA handles data temperatures. It stores the hot data in memory (most expensive). Warm data is stored in the persistent disk layer (less expensive than memory). Cold data can be stored outside of SAP HANA in data archives or near-line solutions (lowest cost). For now, we will focus on hot and warm data.

It can be difficult to decide what is hot and what is warm data, and how the data moves between these layers. SAP HANA can now handle these dilemmas.

Any data that is accessed by any application always comes from memory. This means that if the table is sitting in the persistent layer, the moment it is needed, the table is automatically loaded to memory. It becomes hot data. Column tables can also be partitioned and SAP HANA can intuitively load only the required columns and partitions to memory. It leaves the unwanted columns and partitions in the persistent layer.

You can force data to be loaded from persistence to memory (and the other way around), but it is usually better to allow the applications to handle that based on real usage. Monitoring memory utilization is a key task of an administrator and there are many supplied tools that can support them. What you do not want is continual swapping between memory (hot) and disk (warm), but this happens if memory is not sized large enough to cope.

Dynamic Tiering

We already know that the primary image of a table in SAP HANA is memory-based. This means that applications read tables from memory. If the table is not already loaded to memory (because it was displaced or never loaded), then the table is automatically loaded to memory. This is how active and inactive data is handled. however, there is another option to consider; dynamic tiering.

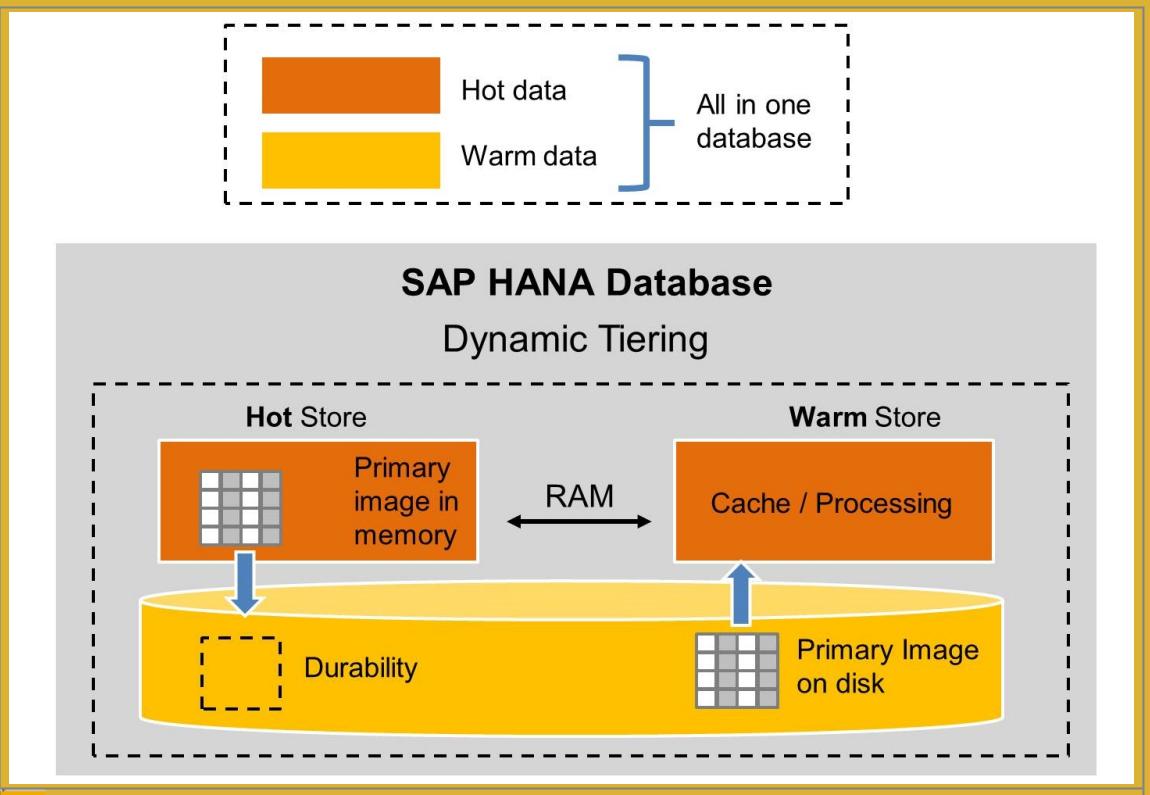


Figure 41: Dynamic Tiering

Since SAP HANA SPS09, you can implement an optional, additional database component known as extended storage. Extended storage is a disk-based storage layer that sits side by side with the existing HANA database. It is used to store tables that do not require the hyper-performance of memory-based tables. Extended storage tables work seamlessly with memory-based tables. A developer or consumer does not need to be concerned (apart from the performance aspect) with where the tables reside.

When an SAP HANA table is first created, you can decide whether it should be memory-based or it should be part of extended storage. This decision can be changed anytime, either manually or controlled by the application. When data becomes more critical, tables can be moved from extended storage to memory (and the other way around).

All tables, whether in memory or extended storage, are listed in the same SAP HANA database catalog. SAP supplies tools to allow administrators to easily monitor both storage areas.

From an architectural perspective, extended storage provides an alternative (or compliment) to Hadoop for Big Data requirements. It can be used to store data that is not subject to strict Service Level Agreements (SLA) and does not need the hyper-performance of memory-based tables.

Technically, extended storage is a column-based database, which is optimized for high-read performance. It borrows much of its technology from SAP IQ database. Although SAP HANA's memory is needed to process the extended storage data, the primary image (the complete table) is not stored in memory, and only data that is needed for each query request passes through memory.

Note:
Extended Storage is the technical name for Dynamic Tiering and is not part of core SAP HANA. It is an SAP HANA option and currently requires a separate license.

Multistore Tables

With SAP HANA 2.0, yet another option exists to support performance of the database — multistore tables.

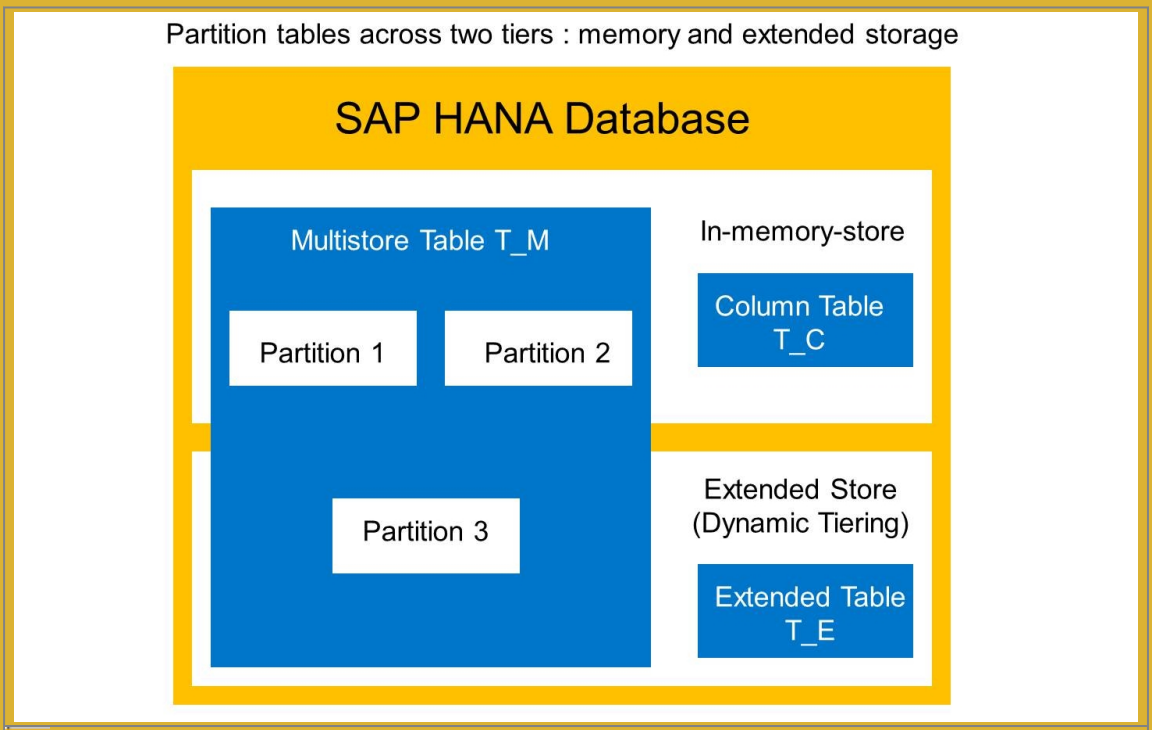


Figure 42: Multistore Tables

Multistore tables are tables that have been partitioned across two tiers — memory and extended storage (you must install extended storage to support multistore tables).

Take for example, a huge table where only the recent data in the table needs to support a strict SLA. The older data does not have to meet the SLA. Since SAP HANA 2.0, you can now have recent partitions in memory, and the remaining partitions on disk in extended storage. Before SAP HANA 2.0 you could not split partitions across the two tiers. They would either all be in memory or all be on disk in extended storage. Now you have the best of both worlds.

Active/Active-Read Enabled Mode

Since SAP HANA 1.0 SPS11, it has been possible to replicate the database log from HANA to a secondary instance of HANA and then to asynchronously replay the database log in the secondary system. This keeps the two HANAs in sync. The reason for doing this is to provide a hot, continuous backup to use in case of primary system failure. You can easily switch to the secondary system to continue with almost no disruption. Many customers have implemented this hot-standby solution.

Since SAP HANA 2.0, we have introduced active/active-read enabled mode, which builds on the solution described previously.

Active/Active-Read Enabled Mode

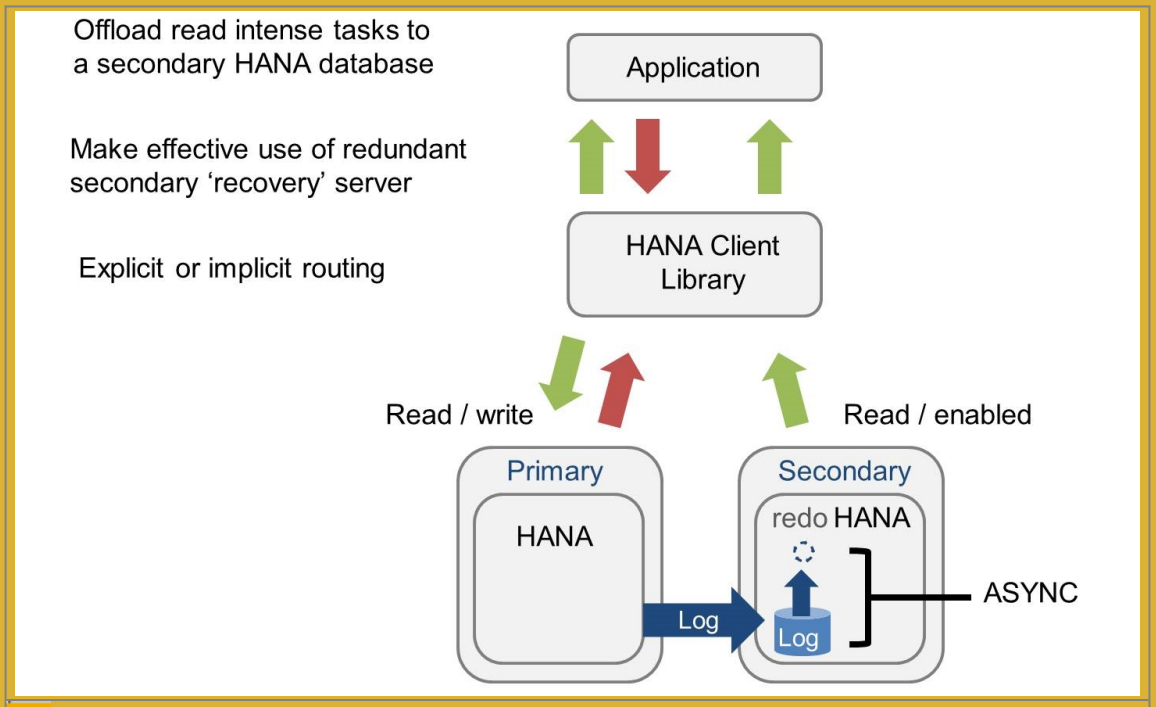


Figure 43: Active/Active-Read Enabled Mode

With active/active-read enabled mode, you can actually use the fully-synced, secondary HANA system for read-intensive tasks. This improves the overall performance of the SAP HANA database. It does this by providing a better balance of workloads, as read-intense workloads can now instead be read from the secondary system.

Applications can now be hard-coded to explicitly redirect read-intense activities to the secondary system, and write activities to the primary system (the two systems are automatically kept in sync). Applications can also use implicit hint-based routing to determine where the read should take place. This means that you do not hard code the routing, but allow the application to determine this dynamically.

SAP HANA Cockpit provides a side-by-side view of the performance of both systems for monitoring purposes.

Maintaining Good Read Performance with Frequent Database Updates

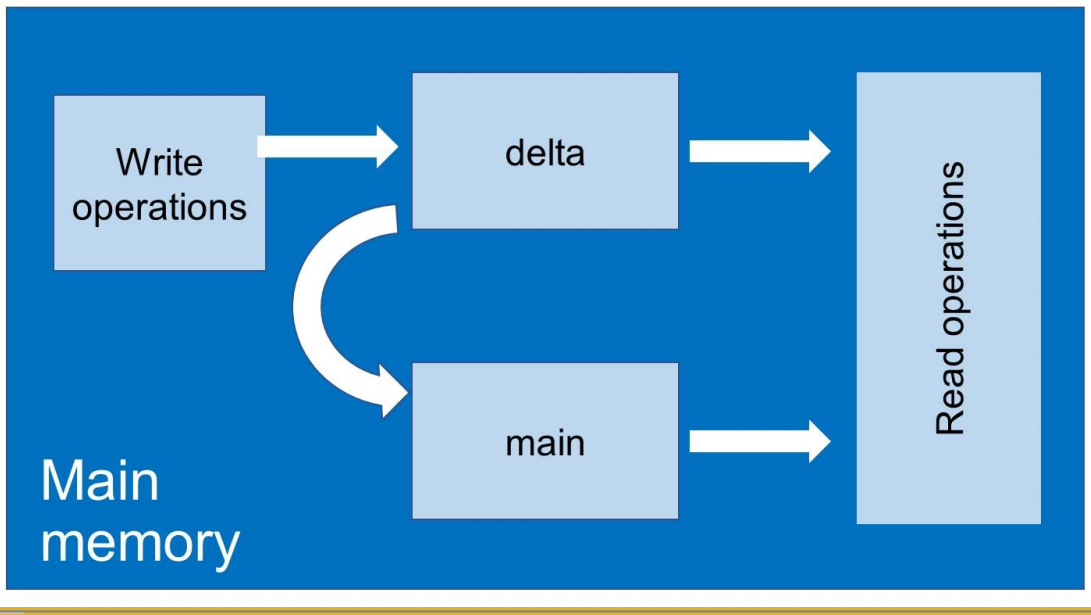


Figure 44: Delta Merge

Updating and inserting data into a compressed, sorted column store table is a costly activity. This is because each column has to be uncompressed, the new records are inserted and then recompressed again. Thus, the whole table is reorganized each time.

For this reason, SAP has separated these tables into a Main Store (read-optimized, sorted columns) and Delta Store (write-optimized, non-sorted columns or rows). There is a regular automated database activity that merges the delta stores into the main store. This activity is called Delta Merge.

Queries always run against both main and delta storage simultaneously. The main storage is the largest one, but because its data is compressed and sorted, it is also the fastest one. Delta storage is very fast for insert, but much slower for read queries, and therefore kept relatively small by running the delta merge frequently.

The delta merge can be triggered based on conditions that can be set. For example, you could define a condition that checks if the delta size is greater than 5% of the main size. If so, the delta merge is triggered.

Delta merge can also be triggered by an application. For example, SAP BW is able to suggest to SAP HANA that after a large data load, a delta merge would be a good idea. Staying on top of the delta merge is critical to maintaining good performance of SAP HANA and the administrator is responsible for this.

Note:
Refer to training course HA200 to learn more about delta merge.

Multitenancy

SAP HANA can run multiple, isolated applications within the same system. This is called multi-tenancy. The official name given to this concept by SAP is Multi-Database Containers (MDC) or more recently this has been changed to simply Multi-Tenancy (the word ‘container’ was used in many places so SAP decided to remove this word from the multi-tenancy concept).

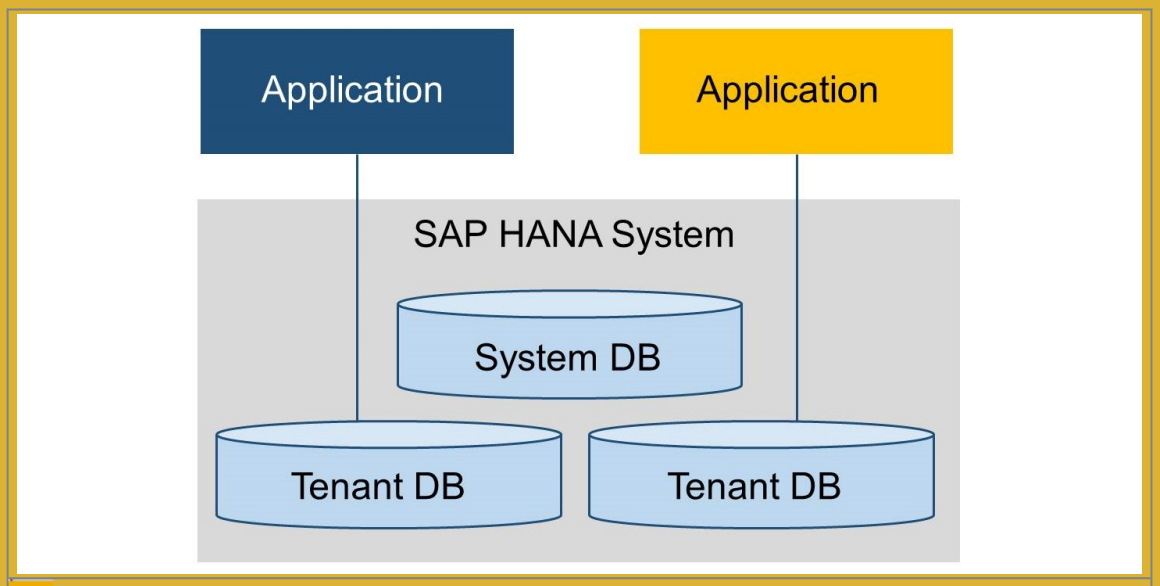


Figure 45: Multitenancy

With multitenancy, there is a strong separation of business data and users who must be kept apart. Each tenant has its own isolated database. Business users would have no idea that they are sharing a system with others running different applications.

The System DB layer is used to manage the system-wide settings and cross-tenant operations, such as backups.

The benefit of a multitenancy platform is that we can host multiple applications on one single SAP HANA infrastructure and share common resources to simplify and reduce costs. But each application is completely isolated from a data perspective but also from a resource perspective. This means each application is assigned its own chunk of SAP HANA processing power and this does not interfere with other applications.

Multitenancy is the basis for cost-efficient cloud computing.

Multi-tenancy in the cloud?

There is no need for SAP to provide multi-tenancy for the SAP HANA cloud service because the same outcome can be achieved by simply deploying new instances of SAP HANA in the cloud. Each instance is completely isolated and can be sized as required.

Catalog vs Container Approach for Database Object Management

When SAP HANA 1.0 was released, all references from the data models and application code to the database objects (e.g. tables and functions) needed to include the schema name. So if a table was needed in an SQL statement it would look like this:

```
SELECT * FROM "SCHEMA A.SALES"
```

This approach meant that if a table was moved to a different schema the code would fail. Hard-coding schemas to code and data models is also a problem when it comes to supporting multiple deployments of the same application. How could you have different versions of a table using this catalog/schema approach? You would have to manually create separate schemas in the catalog with different versions of the tables in each schema, and then have two version of the application code, each pointing to the different schemas. This is called the **catalog** approach and although still supported in SAP HANA 2.0, it is no longer the recommended approach.

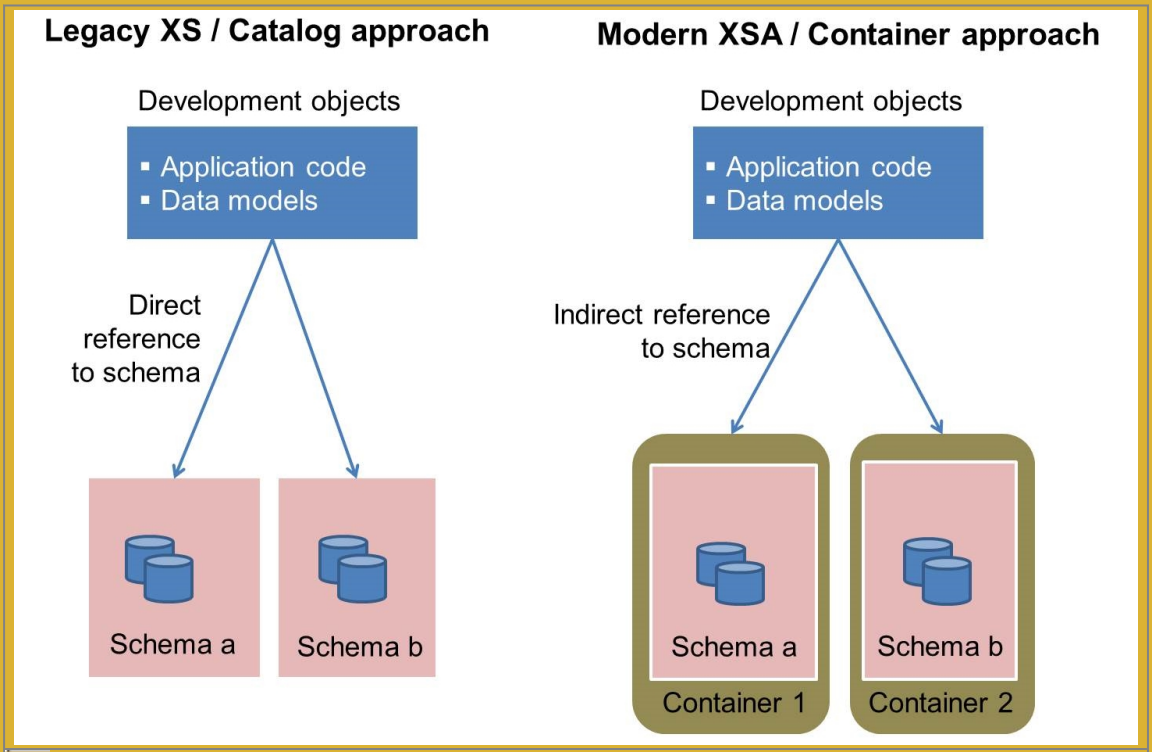


Figure 46: Catalog vs Container Approach

. Since SAP HANA 1.0 SPS11, SAP have introduced a completely new approach to managing multiple deployments of an application and removed the need for fixed schema references.. Instead of referring directly to database schemas, we now write code and develop data models 'schema-less'. What this means is that a developers never needs to worry about the physical schema name in which their database objects sit. All their development now refers to a database **container**. A container is automatically generated the first time you build your application. A container is a logical layer that sits above the physical schema. You can deploy an application multiple times and each deployment can generate a new container to ensure isolation of database objects. So database tables can exists at different versions. Schemas still exist and are used to manage the physical storage of database objects, but they are hidden from the developer. A container always refers to one main schema where the database objects are stored, but a container also refers to other generated schemas that are used for general management of an application (in other words, these extra schemas are used internally and are of no interest to a developer).

So now the SQL code would look like this:

```
SELECT * FROM SALES
```

Notice there is no reference to a fixed schema any longer? As we move the application code to a new version and a new container is generated during the build of the application, the code follows the location of the new table. The container is a key component in the new **HDI** approach to building modern, multi-target applications (MTA).

Database artifacts and source code would sit in a giant single system-wide repository that was organized by packages. From there, development artifacts would be activated, and the generated run-time database objects and applications would be placed in database schemas and in the XS engine run-time. Only one version of the activated objects could exist, which meant you could not deploy an updated version of an application whilst still retaining the original version. XS was tightly integrated with the HANA database, which meant you could not scale either separately.

Each application that was developed could only have one deployment target. For example, you could not develop a single application and deploy it as both an on-premise and a cloud application. You would have to create two separate applications. This original architecture is known as “XS classic/repository based”.

Since SAP HANA 1.0 SPS11, a new, flexible approach that is much more powerful has been introduced for development and run-time. Customers are encouraged to move over to the new architecture to take advantage of many new features. Eventually, XS classic will no longer be relevant and will not be the focus of any more development by SAP.

The catalog approach is tied to XS developments and is supported by SAP HANA Studio. It is possible to view the database objects using the catalog approach with Web IDE for SAP HANA (but this is only to provide a view of the legacy architecture). The container approach is tied to XSA developments and can only be used with Web IDE for SAP HANA, and not SAP HANA Studio.

Unit 2
Exercise 4

Explore the SAP HANA Database with Studio

83

Exercise Objectives

After completing this exercise, you will be able to achieve the following using SAP HANA Studio:

- Locate a table in the HANA database.
- Display the metadata of a table.
- Identify the number of records in a table.
- Preview the content of a table.

Business Example

You are an IT system administrator. You need to explore the SAP HANA database using SAP HANA Studio.

During this exercise, use the following tables from the *TRAINING* schema:

- MARA

  This is a table that contains the product master data in an SAP ERP database.

- HANA_SEARCH_DATA

  This is a table of selected customers and their information.

1. Start the SAP HANA Studio and log on to the SAP HANA system. You can skip this step if you have already logged on.

   For this exercise, you can use either the *SAP HANA Administration Console* perspective or the *SAP HANA Modeler* perspective.

2. Locate the table *MARA* by using a filter on the *Tables* node.
3. Open the definition of table *MARA* and identify whether the table is row or column store.
4. Identify the key columns of the table *MARA*.
5. Identify the number of records loaded to the table and also the storage used by the main and delta areas.
6. Preview the data of table *MARA*.
7. Open the definition of table *HANA_SEARCH_DATA* and identify the table storage type and also determine why there is no delta storage value for this table and why there are no partitions available.
   There is no *Partitions* area of this view and you will not find the *Delta Storage* information because partitions and delta storage are not relevant for row tables, only for column tables.

Explore the SAP HANA Database with Studio

84

Exercise Objectives

After completing this exercise, you will be able to achieve the following using SAP HANA Studio:

- Locate a table in the HANA database.
- Display the metadata of a table.
- Identify the number of records in a table.
- Preview the content of a table.

Business Example

You are an IT system administrator. You need to explore the SAP HANA database using SAP HANA Studio.

During this exercise, use the following tables from the *TRAINING* schema:

- MARA

  This is a table that contains the product master data in an SAP ERP database.

- HANA_SEARCH_DATA

  This is a table of selected customers and their information.

1. Start the SAP HANA Studio and log on to the SAP HANA system. You can skip this step if you have already logged on.

   For this exercise, you can use either the *SAP HANA Administration Console* perspective or the *SAP HANA Modeler* perspective.

2. Locate the table *MARA* by using a filter on the *Tables* node.

   a) In the *Systems* view, expand the *Catalog* folder. To view the *Tables* node, locate the *TRAINING* schema.

   b) Right-click the *Tables* node and choose *Filters*.

   c) Enter the filter pattern **MA**, and choose *OK*.

   d) Expand the *Tables* node.

   The table list is now filtered and displays the table *MARA* in the filtered list. Note that the icon for the table represents column store, as this is a column store table.

3. Open the definition of table *MARA* and identify whether the table is row or column store.

   a) Right-click the table *MARA* and choose *Open Definition*.

   This screen shows the table structure with all columns, their data types, and length.

The type of table is column store. This information is provided in the upper-right corner of the screen.

4. Identify the key columns of the table *MARA*.
   a) The key fields of the table are marked in the *Key* column.
      For table *MARA*, the key columns are *MANDT* (client) and *MATNR* (material number).
5. Identify the number of records loaded to the table and also the storage used by the main and delta areas.
   a) To see the number of records, choose the *Runtime Information* tab.
   b) Locate the *Memory Consumption* figures on the left side of the screen.
6. Preview the data of table *MARA*.
   a) Use either of the following methods to do so:
      - Right click the table name and choose *Open Content* to display the table content.
      - Right-click the table name and choose *Open Data Preview* to explore the table contents as raw data (flat table view), analyze distinct values, or build a quick analysis using a chart or a table.
7. Open the definition of table *HANA_SEARCH_DATA* and identify the table storage type and also determine why there is no delta storage value for this table and why there are no partitions available.
   a) Right-click the *Tables* node and choose *Filters*.
   b) Enter the filter pattern **HANA** and choose *OK*.
   c) To view the filtered table list, expand the *Tables* node.
   d) Right-click the table *HANA_SEARCH_DATA* and choose *Open Definition*.
   e) In the top-right of the view, identify the type of the *HANA_SEARCH_DATA* table is *Row Store*.

   There is no *Partitions* area of this view and you will not find the *Delta Storage* information because partitions and delta storage are not relevant for row tables, only for column tables.

87

Explore the SAP HANA Database with Web IDE for SAP HANA

Exercise Objectives

After completing this exercise, you will be able to perform the following tasks in the Web IDE:

- Locate a table in a classic database view
- Locate a synonym in a database container
- Preview the content of a table

Business Example

You are an IT system administrator. You need to explore the SAP HANA database using Web IDE for SAP HANA.

During this exercise, you will locate and explore the table *PRODUCT*. This is the table that contains product master data.

Task 1: Add your generated database container to the Database Explorer

1. If the SAP Web IDE for SAP HANA is not already open you must launch it and log on using the details from the table, *Credentials*. Otherwise, you can skip this step.

   Table 6: Credentials

Field	Value
User	**`STUDENT##`**
Password	**`Training1`**

2. Switch to the *Database Explorer*, and when prompted, confirm that you wish to add a database to the *Database Explorer*. (This action simply opens the pane, so you don't have to press the '+' button, and only appears when the list is empty).
3. Add your database container to the *Database Explorer*. Your database container begins with *STUDENT##*.
4. There is only one table in your container. Locate this table.
5. Explore the synonyms that have been created for you in your database container.

   Note:
   We use some of these synonyms in a later exercise. They point to the tables in the schema *TRAINING* that we all share.

6. View the data for the table *PRODUCT*.

Task 2: Add a classic catalog schema based entry to the Database Explorer

In this step, add an entry to the Database Explorer that resembles the database connection we created in HANA Studio. We do this so you can directly view the database schemas and the objects they contain. This isn't always needed, as with XSA projects we work only with the container and never the schemas and in fact references to schemas from any design artifact should be avoided. But it is often helpful to a developer to have a view of the actual database schemas too so we can see what is generated from our builds. It is also helpful to be able to see what is available in other schemas that we might want to create synonyms for in our container to allow access.

1. Add an *SAP HANA Database (Multi tenant)* to the *Database Explorer* using the information shown in the table, *Database Information*.

   Table 7: Database Information

Field	Value
Database Type	*SAP HANA Database (Multitenant)*
Host	**`wdflbmt7215`**
Instance Number	**`00`**
Database name	**`H00`** (first select Tenant Database radio button)
User	**`STUDENT##`**
Password	**`Training1`**
Name to Show in Display	**`Classic schema view`**

2. Expand your database *Classic schema view* and view the contents of table *PRODUCT*. This is found in the schema *TRAINING*.

You might like to demonstrate how we could grant permissions to the external user (STUDENT##) so that the generated schema and database objects that belong to the containers also can appear in the Database Explorer under the classic (Studio style) view. If you don't do this then the generated schema that belongs to the container, although it exists, does not appear alongside the classic schemas. This is because the STUDENT## user does not have permission to see the container's schema by default, you have to explicitly grant access.

1	In the Database/Containers list on the left, choose the *Classic schema view* entry that you created earlier.
2	Choose the *Open SQL Console* icon.
3	In Windows Explorer, open the file *HA100 → SQL for External Container Access.txt*.
4	Copy and paste the content of this file to the SQL Console in the Web IDE. The code is: `GRANT "HA100_##_HDI_CONTAINER_1::access_role"` to TRAINING_ROLE_##
5	Replace ## with your group number and Execute the SQL.

6	You can test if this worked because under the catalog node of your *Classic schema view* you should now see your container's schema *HA100_##_HDI_CONTAINER_1*, and the database objects it contains.

89

Explore the SAP HANA Database with Web IDE for SAP HANA

Exercise Objectives

After completing this exercise, you will be able to perform the following tasks in the Web IDE:

- Locate a table in a classic database view
- Locate a synonym in a database container
- Preview the content of a table

Business Example

You are an IT system administrator. You need to explore the SAP HANA database using Web IDE for SAP HANA.

During this exercise, you will locate and explore the table *PRODUCT*. This is the table that contains product master data.

Task 1: Add your generated database container to the Database Explorer

1. If the SAP Web IDE for SAP HANA is not already open you must launch it and log on using the details from the table, *Credentials*. Otherwise, you can skip this step.

   Table 6: Credentials

Field	Value
User	**`STUDENT##`**
Password	**`Training1`**

   a) Double-click the file *Web IDE for SAP HANA* in the folder *Favorites* → *HA100* → *URLs*.

   b) At the logon screen, enter the credentials from the table, *Credentials*, and choose *Log On*.

   c) Close the *Tips and Tricks* dialog box that appears.

2. Switch to the *Database Explorer*, and when prompted, confirm that you wish to add a database to the *Database Explorer*. (This action simply opens the pane, so you don't have to press the '+' button, and only appears when the list is empty).

   a) Choose the  Database Explorer icon on the left of the screen.

   b) At the prompt, choose *Yes*..

3. Add your database container to the *Database Explorer*. Your database container begins with *STUDENT##*.

a) Make sure the *Database Type* selector is set to *HDI Container*.

b) In the list of HDI containers, locate and highlight your container.
Your container is the one that begins with *STUDENT##*.

c) For the *Name to Show in Display*, overwrite the default name with **`My HA100 Container`**.

d) Choose *OK*.
After a few moments your container should appear on the top-left pane.

4. There is only one table in your container. Locate this table.
   a) Expand your container and you should see a list of database objects types.
   b) Choose *Tables*.
   You should see, in the lower pane, a single table with the name *Employees_Gender_00*. This is an empty table.
5. Explore the synonyms that have been created for you in your database container.

> Note:
> We use some of these synonyms in a later exercise. They point to the tables in the schema *TRAINING* that we all share.

   a) In the list of object types, choose *Synonyms*.
   You should then see, in the lower pane, a list of all synonyms that expose all tables needed in this training.
6. View the data for the table *PRODUCT*.
   a) Right-click the synonym *HA100 :: PRODUCT* and choose *Open Data*.
   A new tab appears displaying the contents of the table.

Task 2: Add a classic catalog schema based entry to the Database Explorer

In this step, add an entry to the Database Explorer that resembles the database connection we created in HANA Studio. We do this so you can directly view the database schemas and the objects they contain. This isn't always needed, as with XSA projects we work only with the container and never the schemas and in fact references to schemas from any design artifact should be avoided. But it is often helpful to a developer to have a view of the actual database schemas too so we can see what is generated from our builds. It is also helpful to be able to see what is available in other schemas that we might want to create synonyms for in our container to allow access.

1. Add an *SAP HANA Database (Multi tenant)* to the *Database Explorer* using the information shown in the table, *Database Information*.

Table 7: Database Information

Field	Value
Database Type	*SAP HANA Database (Multitenant)*
Host	**`wdflbmt7215`**
Instance Number	**`00`**

Field	Value
Database name	**H00** (first select Tenant Database radio button)
User	**STUDENT##**
Password	**Training1**
Name to Show in Display	**Classic schema view**

a) In the *Database Explorer* view, select the button **+**.

b) In the *Database Type* selector, choose *SAP HANA Database (Multitenant)*.

c) Enter the information shown in the table, *Database Information*.

d) Choose *OK* and the database now appears in the top pane.

2. Expand your database *Classic schema view* and view the contents of table *PRODUCT*. This is found in the schema *TRAINING*.

a) In the database *Classic schema view*, expand the *Catalog* node.

b) Expand the schema *TRAINING* and choose *Tables* .

c) Right-click the table *PRODUCT* and choose *Open Data*.
This is the same table you viewed in the previous step via a synonym.

You might like to demonstrate how we could grant permissions to the external user (STUDENT##) so that the generated schema and database objects that belong to the containers also can appear in the Database Explorer under the classic (Studio style) view. If you don't do this then the generated schema that belongs to the container, although it exists, does not appear alongside the classic schemas. This is because the STUDENT## user does not have permission to see the container's schema by default, you have to explicitly grant access.

1	In the Database/Containers list on the left, choose the *Classic schema view* entry that you created earlier.
2	Choose the *Open SQL Console* icon.
3	In Windows Explorer, open the file *HA100 → SQL for External Container Access.txt*.
4	Copy and paste the content of this file to the SQL Console in the Web IDE. The code is: `GRANT "HA100_##_HDI_CONTAINER_1::access_role"` to TRAINING_ROLE_##
5	Replace ## with your group number and Execute the SQL.
6	You can test if this worked because under the catalog node of your *Classic schema view* you should now see your container's schema *HA100_##_HDI_CONTAINER_1*, and the database objects it contains.

LESSON SUMMARY

You should now be able to:

- Understand key features of SAP HANA database

Describing High Availability

93

LESSON OVERVIEW
In this lesson we will explain how SAP HANA keeps running during power outages.

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Describe high availability

SAP HANA and Managing Server Failure

SAP HANA utilizes memory for storage, but once the power is gone, all of the data in memory is lost.

SAP HANA must ensure that you do not lose data when the power goes.

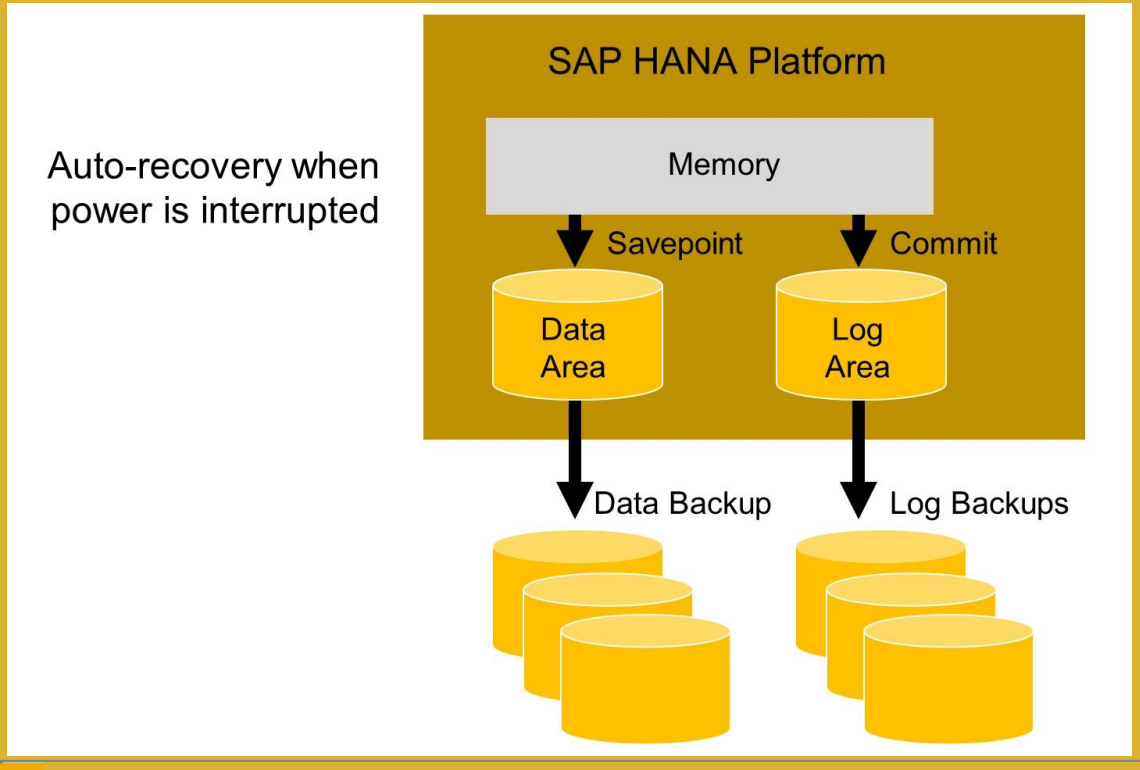


Figure 47: High Availability: Power Interruption

Every few minutes, SAP HANA automatically takes a snapshot of the entire memory. It stores this snapshot on the disk layer in an area called the data volume. This is called a savepoint. The frequency of savepoints is configurable and it depends on how frequently the database changes due to updates, inserts, and deletes. When power is restored, SAP HANA

automatically reloads memory from the last savepoint. It is possible to collect many savepoints over time so that a restore can take place from any point in time.

However, it is important to develop a mechanism to ensure that no data is lost, even between savepoints. To do this, every committed transaction is recorded and saved to a log area. This log area is often based on flash memory (SSD) to ensure ultrafast access. Thus, every update to the database since the last savepoint is captured.

High Availability: Power Interruption

When power is restored, SAP HANA automatically reads the last savepoint and then reapplies the committed transactions from the log since the savepoint. This ensures that the system is exactly where it was at the precise moment you lost the power.

This all happens automatically in the background.

Even though a complete reload of memory from the persistent layer is fast, it is also possible to identify the most important tables so that they are loaded to memory ahead of the tables that are less important. This allows you to restore great performance as quickly as possible.

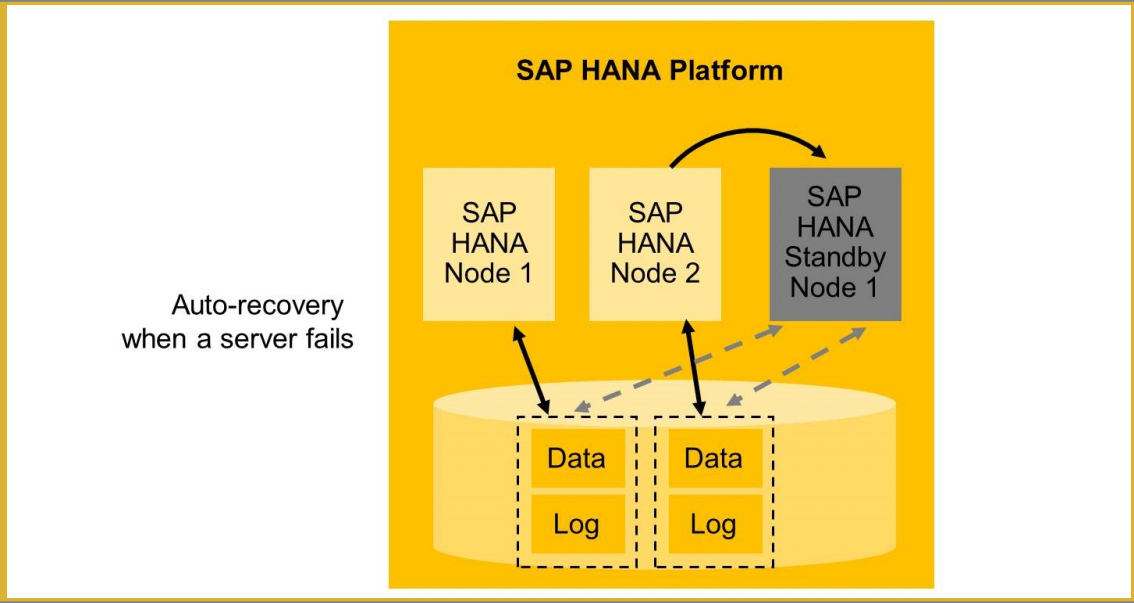


Figure 48: High Availability: Server Down

SAP HANA can be installed across multiple nodes. This is called scale-out. Scale-out is often used to spread the processing load across multiple servers, which improves performance. Scale-out is also used to provide redundant servers that are on standby in case active servers fail.

If a server fails, SAP HANA can automatically swap out to a standby server to ensure that downtime is minimized, or even eliminated.

A standby server can be on warm standby, which means that it is in a near-ready state and does not need to be started from cold.

Standby Servers

Standby servers can also be on hot standby. In this case, the primary server replicates the database log in real-time to a secondary server. This secondary server continuously replays the database log so that the databases are always in sync. This means that there is almost no downtime when switching to the standby server, as it is identical to the primary server at all

times. This approach would be necessary for a mission-critical operation where down-time would be harmful to the business.

Note:
The secondary server can also be used in an active/active-read enabled mode. This means that not only is the secondary server used in case the primary server fails, but the secondary server is also used to offload all read-intense work away from the primary server to balance the workloads.

For standby servers that are not running in hot standby mode, SAP HANA uses the savepoints and logs, described earlier, to load the standby server with the latest data. This means that more time is taken to bring the servers up than hot standby.

So, for mission critical applications, and where SLAs are implemented, you can ensure customers' systems are always running by implementing these fail-over solutions.

Note:
For more information on all high-availability, refer to course HA200.

LESSON SUMMARY
You should now be able to:

- Describe high availability

Learning Assessment

97

1. Which is a valid tool for extracting, transforming and loading data to SAP HANA, and is built in to the SAP HANA platform?

   Choose the correct answer.

   - [X] A Smart Data Integration (SDI)
   - [X] B SAP Data Services
   - [X] C SAP Landscape Transformation Replication Server (SLT)

2. What is used to deliver the different levels of capabilities of SAP HANA 2.0?

   Choose the correct answers.

   - [X] A Editions
   - [X] B Support Pack Stacks (SPS)
   - [X] C Options

3. SAP Enterprise Architecture Designer is used to generate complete SAP HANA based applications.

   Determine whether this statement is true or false.

   - [X] True
   - [X] False

4. What are the two main views in the Web IDE for SAP HANA?

   Choose the correct answers.

   - [X] A Database Explorer
   - [X] B Development
   - [X] C Catalog

5. What is a role of the SAP HANA Cockpit?

   Choose the correct answer.

   - [X] A To provide KPIs to help focus on business performance
   - [X] B To provide key system performance information
   - [X] C To provide an application developer collaboration hub

6. What are advantages of column store tables versus row store tables?

   Choose the correct answers.

   - [X] A Data footprint is automatically reduced through compression
   - [X] B Only the columns required for processing are actually loaded to memory
   - [X] C Columns can be partitioned to improve performance
   - [X] D They are optimized for fast writes and updates to records

7. Why do we still need a disk layer when SAP HANA is referred to as an in-memory database?

   Choose the correct answers.

   - [X] A To store data that has been unloaded from memory when memory becomes full
   - [X] B To hold the delta store for newly-arrived records
   - [X] C To enable full database recovery in the event of a power failure
   - [X] D To store data that is frequently used

8. What is a multistore table?

   Choose the correct answer.

   - [X] A A table that can hold data in row and column orientation
   - [X] B A table that can distribute its partitions across memory and extended storage tiers
   - [X] C A table that can be shared across multiple instances of HANA

9. Active/active-read enabled mode allows applications to redirect read-intense workloads to a secondary redundant read-only HANA node?

   Determine whether this statement is true or false.

   - [X] True
   - [X] False

10. What are scale-out scenarios with SAP HANA?

Choose the correct answers.

- [X] A Provision of extra redundant standby servers to be switched to in case of hardware failure
- [X] B Access to remote servers to store archived data that is rarely used
- [X] C Deployment of low-cost generic servers to support high volume data streaming applications
- [X] D Use of multiple servers to spread data storage and processing in order to improve performance

Learning Assessment - Answers

100

1. Which is a valid tool for extracting, transforming and loading data to SAP HANA, and is built in to the SAP HANA platform?

   Choose the correct answer.

   [X] A Smart Data Integration (SDI)

   [X] B SAP Data Services

   [X] C SAP Landscape Transformation Replication Server (SLT)

   Correct!

2. What is used to deliver the different levels of capabilities of SAP HANA 2.0?

   Choose the correct answers.

   [X] A Editions

   [X] B Support Pack Stacks (SPS)

   [X] C Options

   Correct!

3. SAP Enterprise Architecture Designer is used to generate complete SAP HANA based applications.

   Determine whether this statement is true or false.

   [X] True

   [X] False

   Correct!

4. What are the two main views in the Web IDE for SAP HANA?

   Choose the correct answers.

   - [X] A Database Explorer
   - [X] B Development
   - [X] C Catalog

   Correct!

5. What is a role of the SAP HANA Cockpit?

   Choose the correct answer.

   - [X] A To provide KPIs to help focus on business performance
   - [X] B To provide key system performance information
   - [X] C To provide an application developer collaboration hub

   Correct!

6. What are advantages of column store tables versus row store tables?

   Choose the correct answers.

   - [X] A Data footprint is automatically reduced through compression
   - [X] B Only the columns required for processing are actually loaded to memory
   - [X] C Columns can be partitioned to improve performance
   - [X] D They are optimized for fast writes and updates to records

   Correct!

7. Why do we still need a disk layer when SAP HANA is referred to as an in-memory database?

   Choose the correct answers.

   - [X] A To store data that has been unloaded from memory when memory becomes full
   - [X] B To hold the delta store for newly-arrived records
   - [X] C To enable full database recovery in the event of a power failure
   - [X] D To store data that is frequently used

   Correct!

8. What is a multistore table?

   Choose the correct answer.

   - [X] A A table that can hold data in row and column orientation
   - [X] B A table that can distribute its partitions across memory and extended storage tiers
   - [X] C A table that can be shared across multiple instances of HANA

   Correct!

9. Active/active-read enabled mode allows applications to redirect read-intense workloads to a secondary redundant read-only HANA node?

   Determine whether this statement is true or false.

   - [X] True
   - [X] False

   Correct!

10. What are scale-out scenarios with SAP HANA?

    Choose the correct answers.

    - [X] A Provision of extra redundant standby servers to be switched to in case of hardware failure
    - [X] B Access to remote servers to store archived data that is rarely used
    - [X] C Deployment of low-cost generic servers to support high volume data streaming applications
    - [X] D Use of multiple servers to spread data storage and processing in order to improve performance

    Correct!

UNIT 3 Analytical Processing with SAP HANA

UNIT OBJECTIVES

- Understand SAP HANA core modeling
- Describe SAP HANA Live
- Describe SAP S/4HANA Embedded Analytics
- Advanced Modeling with SAP HANA

Understanding Core Modeling with SAP HANA

104

LESSON OVERVIEW
In this lesson we will cover modeling with SAP HANA.

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Understand SAP HANA core modeling

Modeling in SAP HANA

This lesson explains why modeling in SAP HANA is an important activity that ensures you are using the SAP HANA platform to its full potential.

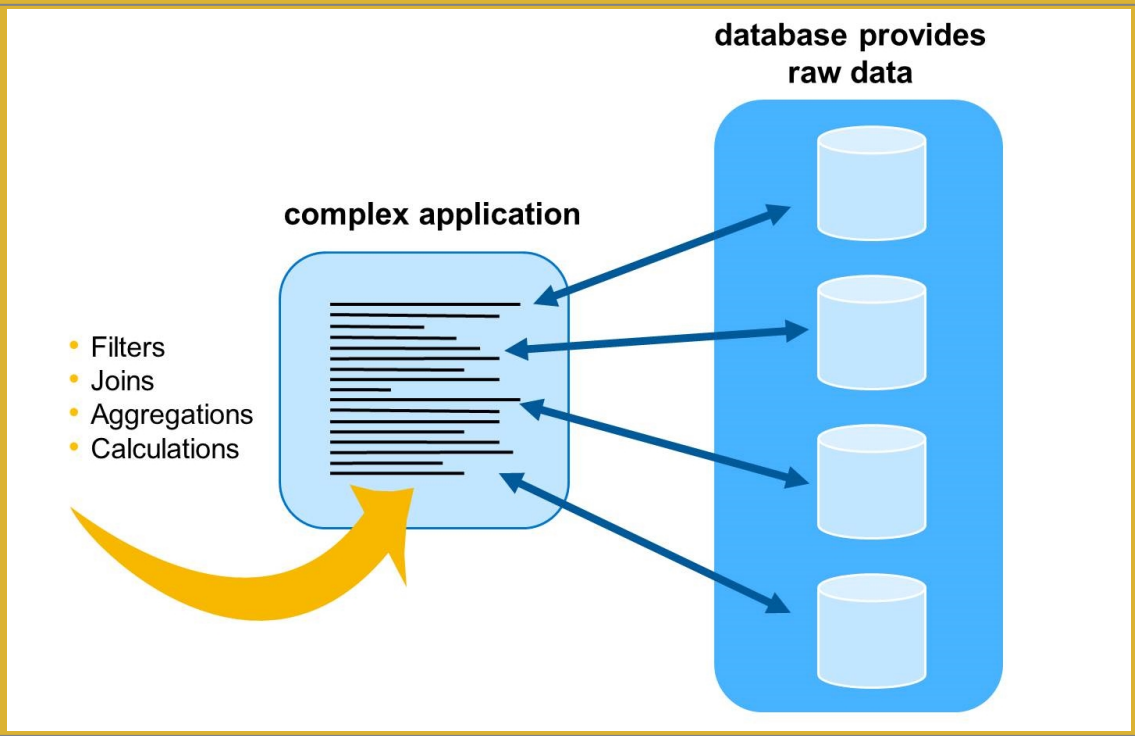


Figure 49: Modeling in the Application

In a traditional application, the role of the database is to provide raw data. There is usually very little, or even no data processing in the database.

The raw data is sent from the database directly to the application. The application then begins to process the data by combining it, aggregating it, and performing calculations in order to generate something meaningful.

We can find ourselves moving a lot of raw data between the database and the application. When we move raw data to the application layer we make the application code very complex.

This is because It has to deal with the data processing and modeling tasks as well as managing all of the other process flow control, business logic, User Interface (UI) operations, integrating data from multiple sources, and so on.

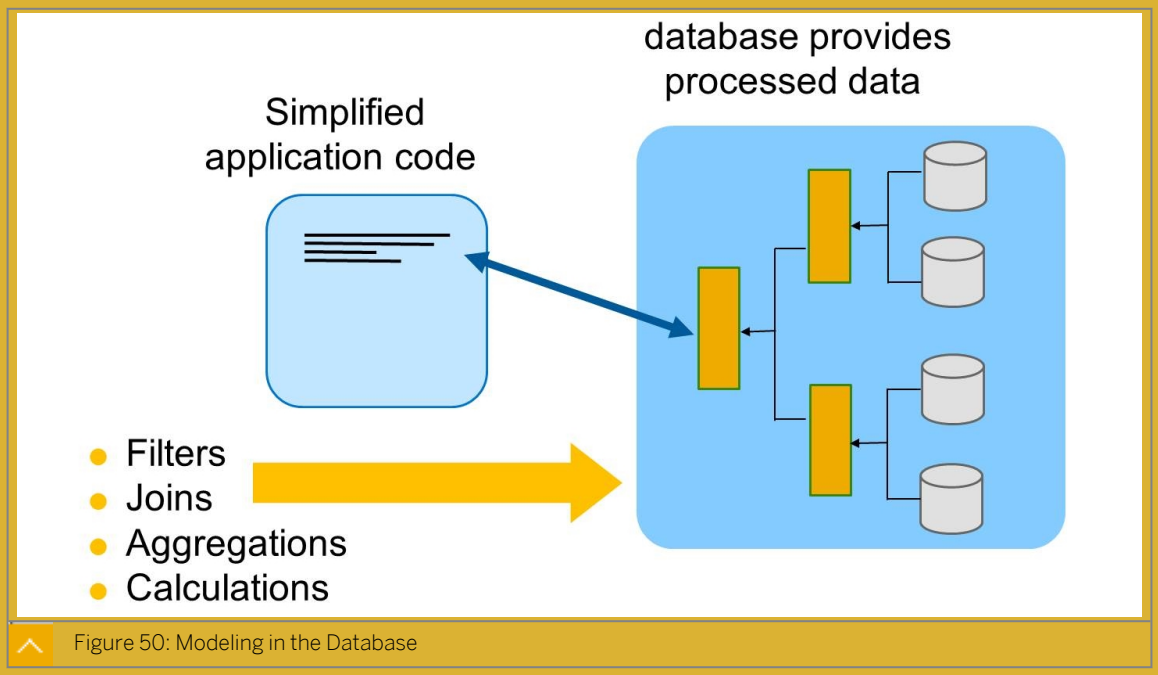


Figure 50: Modeling in the Database

With SAP HANA, we can build a sophisticated modeling layer on top of the database tables. This means we can first process the raw data and turn it into something meaningful in the database before passing it on to the application.

With SAP HANA, we build calculation views to combine data from multiple tables and apply filters, conditions, calculations, and aggregations. The calculation views are developed in SAP HANA using easy-to-use modeling tools, and are stored in SAP HANA alongside the database tables in the database.

Therefore, instead of the application processing the raw data, the application calls the required calculation views and the processing is pushed down to SAP HANA. This is efficient in the following ways:

- The application code is simplified, as it does not have to deal with many data processing tasks. These tasks are pushed down to SAP HANA where in-memory processing takes place.
- The processing on the data is carried out where the data resides, so we do not have to move raw data from the database to the application. We only move the results of the data processing to the application.
- The calculation views can be reused in multiple applications so we avoid redundancy.

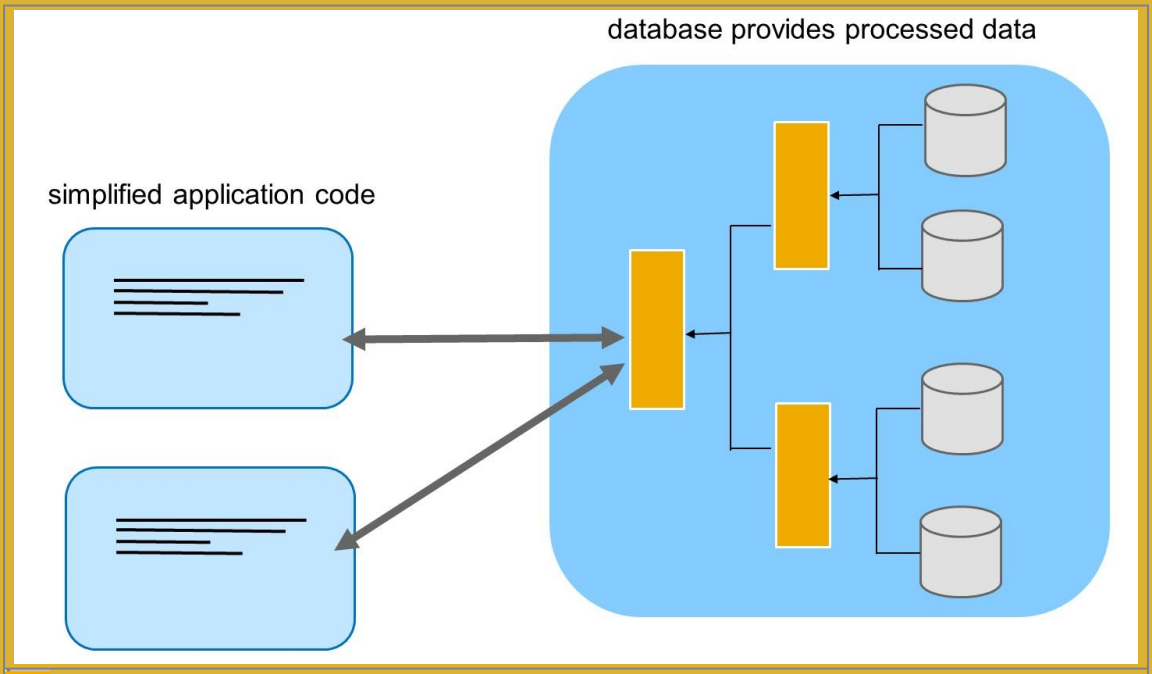


Figure 51: Reuse of Models

In traditional applications, there is a high degree of redundancy in the application code. Developers find themselves continually creating the same code to process data.

When dealing with highly normalized database models, such as those used with SAP Business Suite, there can be many individual tables that need to be called and combined with joins. These joins can often be pushed down to most databases. However, SAP HANA goes beyond helping with just the table joins. SAP HANA can take on the work that was done by the application. SAP HANA takes care of complex calculations and data flow logic, including executing aggregations and disaggregation.

Therefore, you can create an SAP HANA calculation view once and then reuse it. SAP HANA calculation views can contain dynamic placeholders. This means that the applications can pass variables down to the calculation views a response to a filter value that came from a business user. Many of the calculation views can also call procedures that have input parameters.

Calculation views can consume other calculation views. This encourages a high degree of modularization and reuse.

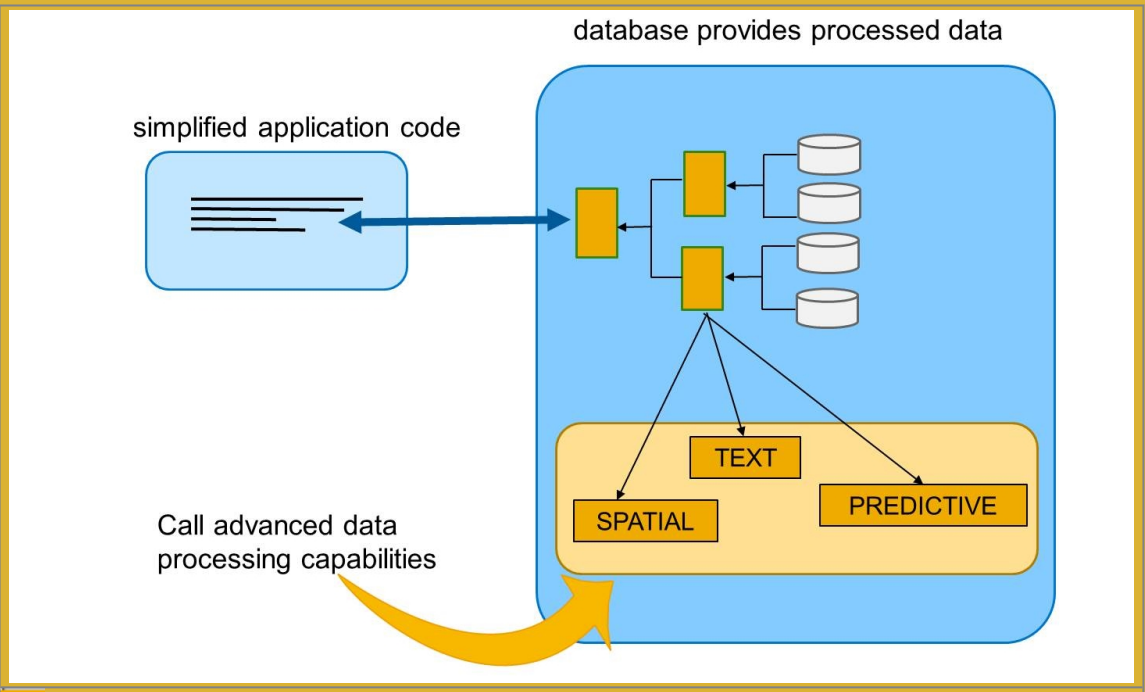


Figure 52: Advanced Data Processing Capabilities

SAP HANA has built-in, advanced data processing capabilities. These include textual, spatial, and predictive functions. Calculation views can easily call these native SAP HANA functions, so applications can leave all the complex processing to SAP HANA.

Core Modeling Versus Advanced Modeling

The term, core modeling (sometimes called view modeling), refers to the development of models that handle common analytical functions. These functions include filtering, aggregation, calculations, and so on.

When we develop models to handle advanced analytical scenarios, such as predictive, spatial, textual, and graph, we refer to this type of modeling as advanced modeling. For now we will focus on core modeling.

Core modeling in SAP HANA begins with the creation of a **calculation view**.

Note:
In earlier releases of SAP HANA, calculations views were part of a family called Information Views. There were originally three members of that family; Attribute View, Analytic View, and Calculation View.

Each of these view types had its own unique features, and typically, all three view types were required. However, since the calculation view has inherited all the features of the two other views, we no longer develop attribute or analytic views. In fact these types of views can be migrated to calculation views using the supplied tools. The calculation view can now do it all, which means we no longer have to be concerned about which view type to use.

Calculation View Creation

When you create a calculation view, you choose various combinations of settings. These settings define four basic types of calculation view.

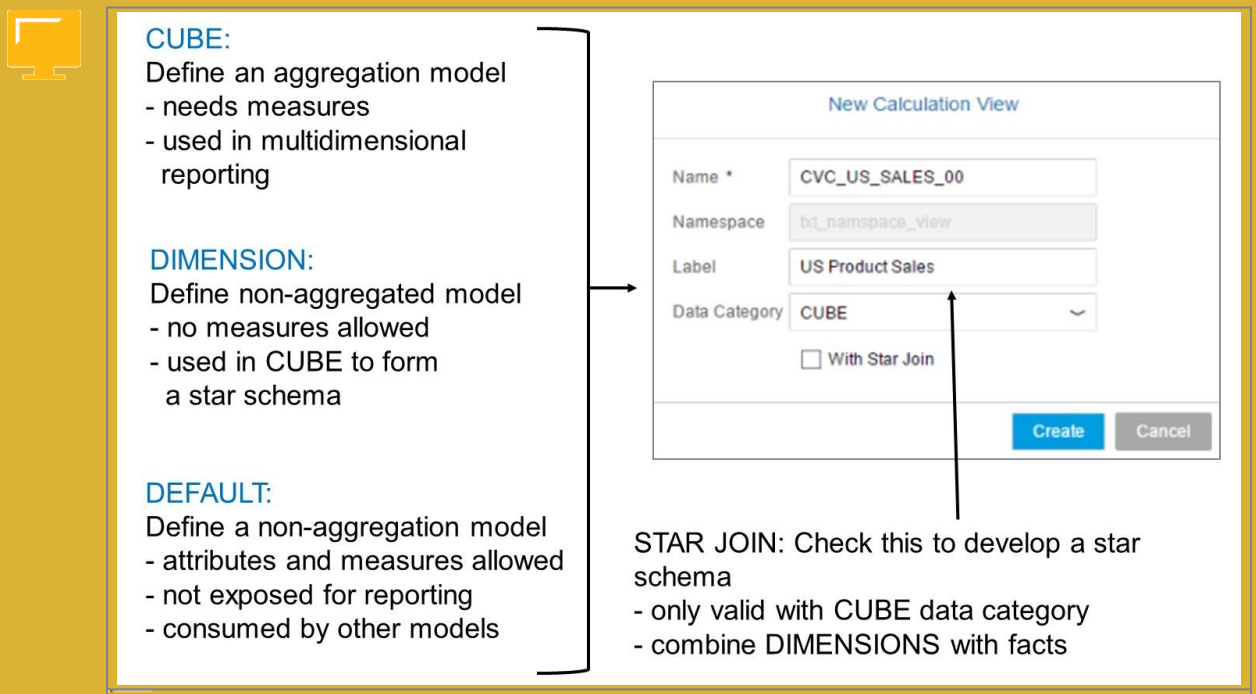
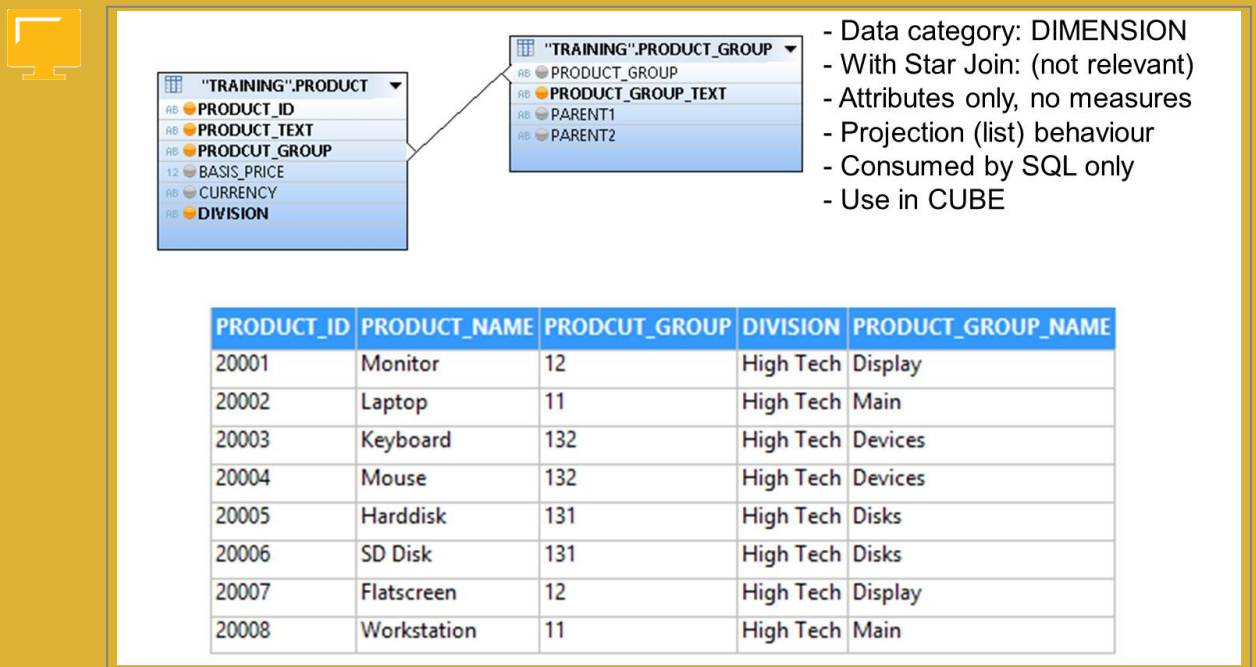


Figure 53: Choosing the Correct Calculation View Settings

The settings are chosen when you first create the calculation view. The four types are as follows:

- Dimension
- Cube without star schema
- Cube with star schema
- Default



PRODUCT_ID	PRODUCT_NAME	PRODCUT_GROUP	DIVISION	PRODUCT_GROUP_NAME
20001	Monitor	12	High Tech	Display
20002	Laptop	11	High Tech	Main
20003	Keyboard	132	High Tech	Devices
20004	Mouse	132	High Tech	Devices
20005	Harddisk	131	High Tech	Disks
20006	SD Disk	131	High Tech	Disks
20007	Flatscreen	12	High Tech	Display
20008	Workstation	11	High Tech	Main

Figure 54: Modeling Dimensions

Dimensions are most likely to be created first. The purpose of a dimension type of calculation view is to define a list of related attributes, such as material, material color, weight, and price. This list can be directly consumed by an application using SQL. However, it is most likely to be found as a component in another calculation view of the type *Cube* when creating star schemas.

Dimension type calculation views cannot contain measures. They can only contain attributes. Without measures, aggregation is not possible. Reporting tools cannot directly access calculation views of type *DIMENSION*. Only direct SQL access is allowed.

It can be helpful to think of calculation views of type dimension as master data views. You would not model transaction data using dimension calculation views as no measures can be defined, and measures are for modeling with transactional data.

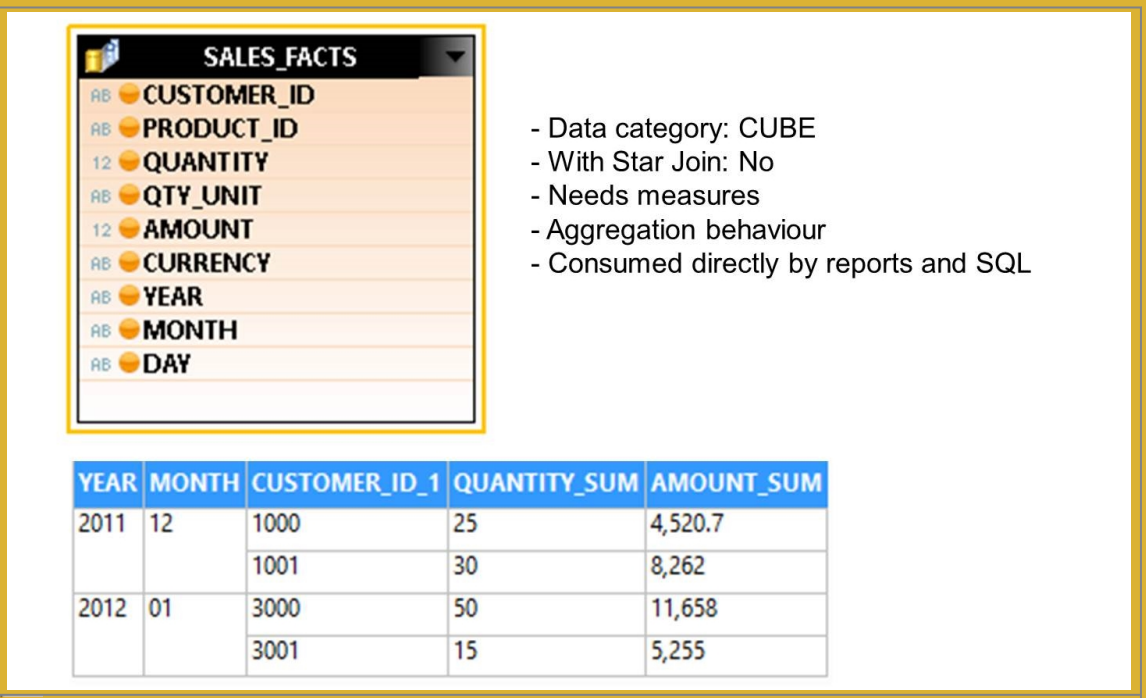
Note:
Be careful not to confuse measures with attributes that are of a numerical data type, such as integer or decimal. A numeric field can be included in this dimension calculation view but it cannot be modeled as a measure (it must be modeled only as an attribute). This means that there is no aggregation behavior possible. For example, you could include weight but you cannot sum this. The output appears as a list of all weights.

Data Category — Dimension

To get started with calculation views of type *DIMENSION*, you need to set the data category to *DIMENSION*.

You then proceed to define the source tables, the joins, the filters, and the columns that are to be exposed. It is also possible to define additional derived attributes. An example of this could be a new column to generate a weight category based on a range of weights, using an `IF` expression.

You are then able to rename any columns to be more meaningful. Remember that the column names originate from the database tables, and these names can often be meaningless to developers and business users.



YEAR	MONTH	CUSTOMER_ID_1	QUANTITY_SUM	AMOUNT_SUM
2011	12	1000	25	4,520.7
		1001	30	8,262
2012	01	3000	50	11,658
		3001	15	5,255

Figure 55: Modeling Cubes

The next type of calculation view is the type *Cube*. This type is used to define a dataset comprised of attributes and measures that can be used in a flexible slice and dice format. This is not a star schema as there are no dimensions defined, but simply a dataset based on one or more transaction tables. These tables can be queried using any combination of the attributes and measures that they include, to create either a line level or an aggregated dataset.

Reporting tools can directly access this type of calculation view. They can also be accessed via SQL.

Data Category — Cube

To get started, set the data category to *CUBE*.

Do not select the *Star Join* flag. This is used later, in the next calculation view type.

You then select the table, or tables, which are to be included in the model. Typically, you choose a transaction table so that you have columns from which you can define attributes and measures. It is possible to include more than one table. For example, you may need to include a header and a line item table to form the complete picture of a sales transaction. In this case, you simply join the tables using a *JOIN* node. You can also merge transaction tables using a *UNION* node.

Then, select the columns from the tables that are to be exposed. You can optionally set filters and define additional calculated columns.

Then, rename any columns to provide meaningful names to the developer and business user.

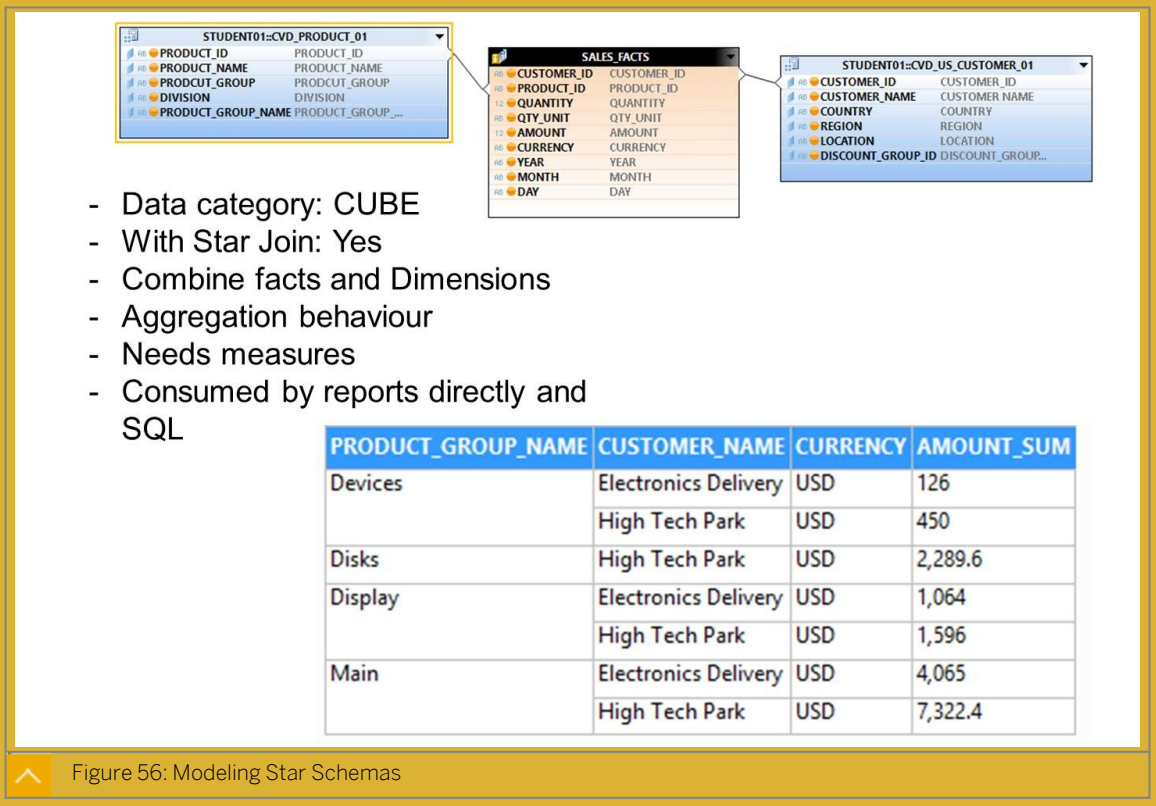


Figure 56: Modeling Star Schemas

The next type of calculation view is the *Cube* type, but with an additional setting — star schema. This is again based on the cube type of calculation view, but with one or more dimension calculation views joined to the model.

Adding the dimension views enables you to request aggregations of any measures in the fact table by any combination of attributes. You are not limited to just those attributes from the fact table, but also attributes from any dimensions. This significantly increases the analysis possibilities.

This type of calculation view follows the same rules as the cube type. It is consumed directly by reporting tools as well as SQL, and it can include attributes and measures. It is used to present aggregated views of the dataset in the most efficient way.

Data Category — Cube Star Join

To get started, make sure that you set the data category to *CUBE* and select the *Star Join* flag.

Select the transaction tables and create joins to combine the transaction tables. Then, choose the columns to expose, set any filters, and create any calculated columns. What you are doing up to this point is forming a fact table that is used as the hub of the star schema.

The next step is to define the star schema by linking relevant calculation views of type *DIMENSION* to the fact table.

Then, improve the names of any columns by using the rename function in the *Semantic* node.

General Purpose Views

The final type of calculation view has the default data category. This type of view is simple and is not meant for multidimensional modeling. It produces flattened result sets that can contain attributes and measures. It is not visible to reporting tools.

These types of views are usually used as data sources to cube type views. This type of view does not expose its metadata to the consuming applications as the other views do. It can help to think of this type of view as an internal view.

SAP HANA Modeling Interfaces

Calculation views can be created in the SAP HANA Studio and also in the Web IDE for SAP HANA. However, a calculation view that is created in Studio is not visible in Web IDE (and vice versa). The reason for this is that each interface client uses a completely different repository framework for the storage of their views and these are not interchangeable. Studio is used for classic XS development and Web IDE is used for XSA development. It is possible to develop models in both environments in the same SAP HANA system, but SAP strongly recommends creating all new modeling artifacts in the newer and more powerful XSA framework. Many customers will be moving from XS to XSA development and during the transition will need to operate both interfaces.

When it comes to developing calculation views, the sequence of steps is similar between the two interfaces and they both produce the same activated calculation view which is deployed as a column view in the database. If you learn modeling in Studio, it is very easy to move your skills to Web IDE.

SAP HANA Studio — Modeler Perspective

With SAP HANA Studio, once you open the *Modeler* perspective, you can create and maintain all types of modeling objects including those that are now deprecated (for example, attribute and analytic views). This is helpful if you need to access these older types of objects. Web IDE cannot provide access to attribute or analytic views and only supports the calculation view (because that is the only type you should be building).

With SAP HANA Studio *Modeler* perspective, you can create the following XS based core modeling objects:

- Attribute View (deprecated)
- Analytic View (deprecated)
- Calculation View
- Decision Table
- Procedure
- Analytic Privilege

You need to use the *Development* perspective to create these:

- Functions
- Flowgraph
- CDS artifacts

Beyond modeling, SAP HANA Studio also offers many other useful functions and tools that a modeler may find helpful. These functions include a flat file import wizard, SLT cockpit, and a helpful *Quick View* pane where many modeler tools and utilities are found in a convenient list. So, even for developers and modelers who use Web IDE, it is possible that they may return to Studio to use some of the tools that are not yet available in Web IDE.

Web IDE for SAP HANA, Modeling Objects

With Web IDE, you can create the following core modeling objects:

- Calculation View
- Procedure
- Function
- Flowgraph
- Analytic Privilege
- CDS artifacts

For developers and modelers, moving from the Studio to Web IDE is straightforward and requires no re-training. All the familiar modeling features are used.

Unit 3
Exercise 6

Create Dimension Calculation Views

115

Exercise Objectives

In this exercise, you learn how to create a calculation view of the type dimension. You will later join this dimension view to the sales fact data in another calculation view of the type cube with star join.

Business Example

You need to make available the following two calculation views in order to provide the master data attributes that are needed for inclusion in other calculation views:

- A calculation view of type dimension, to expose information relating only to US customers.
- A calculation view of type dimension, to expose product master data.

Before starting this exercise, we advise that you preview the content for each of the required tables in order to familiarize yourself with the data and also think about the joins that might be used:

Refer to the course section that covers the Web IDE Database Explorer to learn how to access and browse the tables, and remember you can access these tables via synonyms.

Database Schema: *TRAINING*

Column Tables: *CUSTOMER*, *PRODUCT*, and *PRODUCT_GROUP*.

In this exercise we set a permanent filter for customers in the US only, to provide a simple example of a filter. Please emphasize the other methods of filtering including variables for dynamic evaluation, and also analytic privileges. The key point is that HA100 students do not assume that we must always fix static filters in all calculation views to expose only the data each individual user should see. There are more dynamic ways to do this.

Remember, both calculation views are already created in the solutions folder with the suffix 00. If time is tight or students mess up, you could use one or both of these so the next exercise can still work.

Note:
In this exercise, replace ## with your own student number.

Task 1: Create a calculation view of type dimension for customer master data attributes

1. Start the Web IDE and open the *Development* view.
2. Create a new folder with the name **`exercises`** as a sub-folder under *HA100* → *HDB* → *src*.

3. In your *exercises* sub–folder, create a new calculation view of the type dimension, with the following properties:

   Table 8: Calculation View Properties

Field	Value
Name	**CVD_US_CUSTOMER_##**
Label	**US Customers**
Data Category	*Dimension*

4. Create a new *Projection* node and assign to it the *CUSTOMER* table from the *TRAINING* schema.
5. Add all fields from the *CUSTOMER* table to the output.
6. To show entries only for the US (United States), filter the *COUNTRY* table column.
7. Connect the two *Projection* nodes.
8. Add all columns to the output of the final *Projection* node.
9. Change *CUSTOMER_TEXT* to *CUSTOMER_NAME* for the **name** of the column ('name' is the technical name – no spaces allowed).
10. Change *CUSTOMER_TEXT* to *Name of customer* for the **label** of the column ('label ' is the user-friendly name used in reports).
11. Save and build the new calculation view checking the console to ensure that the calculation view has been successfully built.
12. Preview the data for *CVD_US_CUSTOMER_##*. Check that you only see US customers and that all columns from the source table are shown.

Task 2: Create a calculation view of type dimension for product master data attributes

1. In your *exercises* subfolder, create a new calculation view of the type dimension with the following properties:

   Table 9: Calculation View Properties 2

Field	Value
Name	**CVD_PRODUCTS_##**
Label	**All Products**
Data Category	*Dimension*

2. Add the *PRODUCT* and *PRODUCT_GROUP* tables from the *TRAINING* schema to a new *join* node.
3. Join the column *PRODUCT_GROUP*, from the table *PRODUCT*, to the column *PRODUCT_GROUP*, from the table, *PRODUCT_GROUP*, using an inner join and n..1 cardinality.
4. From the *HA100 :: PRODUCT* table, map only the four fields *PRODUCT_ID*, *PRODUCT_TEXT*, *PRODUCT_GROUP*, and *DIVISION* to the output.

5. From the *HA100 :: PRODUCT_GROUP* table, map only the field *PRODUCT_GROUP_TEXT* to the output.
6. Connect the *join* node to the *projection* node.
7. Add all columns to the output of the final *projection* node.
8. Change *PRODUCT_TEXT* to *PRODUCT_NAME* for both the label and also the name.
9. Save and build the new calculation view. Then check the console to ensure that the calculation view has been successfully built.
10. Preview the data for the calculation view *CVD_PRODUCT_##*. Check that you see the product group text displayed against all products (this proves that the join is working).

Create Dimension Calculation Views

118

Exercise Objectives

In this exercise, you learn how to create a calculation view of the type dimension. You will later join this dimension view to the sales fact data in another calculation view of the type cube with star join.

Business Example

You need to make available the following two calculation views in order to provide the master data attributes that are needed for inclusion in other calculation views:

- A calculation view of type dimension, to expose information relating only to US customers.
- A calculation view of type dimension, to expose product master data.

Before starting this exercise, we advise that you preview the content for each of the required tables in order to familiarize yourself with the data and also think about the joins that might be used:

Refer to the course section that covers the Web IDE Database Explorer to learn how to access and browse the tables, and remember you can access these tables via synonyms.

Database Schema: *TRAINING*

Column Tables: *CUSTOMER*, *PRODUCT*, and *PRODUCT_GROUP*.

In this exercise we set a permanent filter for customers in the US only, to provide a simple example of a filter. Please emphasize the other methods of filtering including variables for dynamic evaluation, and also analytic privileges. The key point is that HA100 students do not assume that we must always fix static filters in all calculation views to expose only the data each individual user should see. There are more dynamic ways to do this.

Remember, both calculation views are already created in the solutions folder with the suffix 00. If time is tight or students mess up, you could use one or both of these so the next exercise can still work.

Note:
In this exercise, replace ## with your own student number.

Task 1: Create a calculation view of type dimension for customer master data attributes

1. Start the Web IDE and open the *Development* view.

   a) Choose the </> *Development* view.

2. Create a new folder with the name **exercises** as a sub-folder under *HA100* → *HDB* → *src*.
   a) In your workspace, expand the folders *HA100* → *HDB* → *src*.
   b) Right click the *src* folder, and choose *New* → *Folder*.
   c) Enter the folder name **exercises**, and choose *OK*.
3. In your *exercises* sub–folder, create a new calculation view of the type dimension, with the following properties:

   Table 8: Calculation View Properties

Field	Value
Name	**CVD_US_CUSTOMER_##**
Label	**US Customers**
Data Category	*Dimension*

   a) In the *Development* view, right-click the *exercises* subfolder and choose *New* → *Calculation View*.
   b) Enter the properties from the table, *Calculation View Properties*.
   c) Choose *Create*.
4. Create a new *Projection* node and assign to it the *CUSTOMER* table from the *TRAINING* schema.
   a) Expand the node selection toolbar (two arrows to the right) so you can see the full names of each node so they are easy to identify.
   b) From the node selection toolbar, click the *Projection* node then click in the empty space at the bottom of the dataflow to add the new node.
      The new *Projection* node appears directly below the existing *Projection* node.
   c) Make sure that the newly added *Projection* node is selected and to open the data source selector, choose the  *Add Data Source* icon that appears on the right of the node.
      If you don't see the icon (and the other action icons to the right) first click in the empty space on the canvas then click back on the node, the icons then appear.
   d) In the *search* field, enter **Cust**.
      A list of data sources that match your search appear.
   e) Highlight the row where the table *CUSTOMER* appears.
   f) Choose *Finish*.
5. Add all fields from the *CUSTOMER* table to the output.
   a) Select the *Projection* node that you just added and when you see the four vertical action icons to the right of the node, select the last one that opens the details pane to the right. Ensure that you are displaying the *Mapping* tab (it is the default) and then right-click the table header *HA100 :: CUSTOMER*, which is found under *Data Sources*.

b) In the context menu of the table header, choose *Add to Output*.

All fields now appear on the right, under *Output Columns*.

6. To show entries only for the US (United States), filter the *COUNTRY* table column.

a) Choose the *Filter Expression* tab.

b) In the *Elements* tab, expand the *Columns*.

c) Click *Country* so it appears in the expression canvas.

d) From the *Operators* tab, choose **=**.

e) Enter **'** (a single quote) and a second, closing single quote automatically appears. Between the quotes, in uppercase, enter **US**.

7. Connect the two *Projection* nodes.

a) Select the *Projection* node that you added so that the action symbols appear to the right.

b) Drag a line between from the arrow symbol and the circle at the bottom of the *Projection* node above.

The *Projection* nodes are now connected.

8. Add all columns to the output of the final *Projection* node.

a) Select the upper-most *Projection* node, which is directly below the *Semantics* node. Ensure that you are displaying the *Mapping* tab and then right-click the header *Projection_1*.

b) In the context menu, choose *Add to Output*.

All fields appear on the right, under *Output Columns*.

9. Change *CUSTOMER_TEXT* to *CUSTOMER_NAME* for the **name** of the column ('name' is the technical name – no spaces allowed).

a) In the *Semantic* node, click inside the *CUSTOMER_TEXT* field, under the *Name* column and enter **CUSTOMER_NAME**.

10. Change *CUSTOMER_TEXT* to *Name of customer* for the **label** of the column ('label ' is the user-friendly name used in reports).

a) In the *Semantic* node, click inside the *CUSTOMER_TEXT* field, under the *Label* column. Enter **Name of customer**.

11. Save and build the new calculation view checking the console to ensure that the calculation view has been successfully built.

a) Choose the *Save* button, or use the menu path *File* → *Save*.

b) Choose the menu path *Build* → *Build Selected Files*.

In the console, which appears at the bottom right of the screen, you see a log. Ensure that you can see that the build completed successfully.

12. Preview the data for *CVD_US_CUSTOMER_##*. Check that you only see US customers and that all columns from the source table are shown.

a) Right-click your calculation view, which appears on the Development objects tree on the left of the screen, and choose *Data Preview*.

b) Select the *Raw Data* tab.

c) Check the *Country* column to ensure that you only see US customers and all six columns from the original database table display.

Task 2: Create a calculation view of type dimension for product master data attributes

1. In your *exercises* subfolder, create a new calculation view of the type dimension with the following properties:

   Table 9: Calculation View Properties 2

Field	Value
Name	**`CVD_PRODUCTS_##`**
Label	**`All Products`**
Data Category	*Dimension*

   a) In the *Development* view, right-click the *exercises* subfolder and choose *New* → *Calculation View*.

   b) Enter the properties provided in the table, *Calculation View Properties 2*.

   c) Choose *Create*.

2. Add the *PRODUCT* and *PRODUCT_GROUP* tables from the *TRAINING* schema to a new *join* node.

   a) From the node selection toolbar, choose the *join* node then choose the empty space at the bottom of the data flow to add the new node.

   The new node appears directly below the existing *projection* node.

   b) Choose the newly added join node. To open the data source selector, choose the 

   Add Data Source icon that appears on the right.

   c) In the *Search* field, enter **`Product`**.

   d) Highlight the rows where the tables *PRODUCT* and *PRODUCT_GROUP* appear.

   e) Choose *Finish*.

   The two tables appear in the canvas side-by-side.

   Note:
   If the columns of the tables do not appear on the canvas, select a different node then return to the join node. The columns appear (this is a Web IDE refresh issue that will be fixed).

3. Join the column *PRODUCT_GROUP*, from the table *PRODUCT*, to the column *PRODUCT_GROUP*, from the table, *PRODUCT_GROUP*, using an inner join and n..1 cardinality.

   a) Expand the details panel of the join node by selecting the small button in the very top-right corner of the view. The detail panel appears on the right of the screen.

b) Drag the column *PRODUCT_GROUP* from the left table (*PRODUCT*) to the *PRODUCT_GROUP* column of the right table (*PRODUCT_GROUP*) so a line is created between them.

c) Carefully select the join line between the columns to open the Properties of the join in the pane below.

d) in the *Join Type* selector, choose *Inner*

e) In the *Cardinality* selector, choose *n..1*.

4. From the *HA100 :: PRODUCT* table, map only the four fields *PRODUCT_ID*, *PRODUCT_TEXT*, *PRODUCT_GROUP*, and *DIVISION* to the output.

a) Make sure that the *join* node that you recently added is still highlighted and choose the *Mapping* tab.

b) Under the header *HA100 :: PRODUCT*, you see a list of all available source fields from this table. Right-click only the column *PRODUCT_ID* and in the context menu, choose *Add to Output*.

The single column appears on the right side, under the *Output Columns* pane.

c) Repeat substep (b) for *PRODUCT_TEXT*, *PRODUCT_GROUP*, and *DIVISION*.

5. From the *HA100 :: PRODUCT_GROUP* table, map only the field *PRODUCT_GROUP_TEXT* to the output.

a) Under the header *HA100 :: PRODUCT_GROUP* you see a list of all available source fields. Right-click *PRODUCT_GROUP_TEXT* and in the context menu, choose *Add to Output*.

The field appears on the right side, under the *Output Columns* pane. You now have five output fields.

6. Connect the *join* node to the *projection* node.

a) Choose the *join* node (the one you created) so that the action symbols appear to the right.

b) Drag a line between the arrow symbol and the circle at the bottom of the *projection* node above.

The nodes are now connected.

7. Add all columns to the output of the final *projection* node.

a) Choose the *projection* node. Ensure that you are displaying the *Mapping* tab and right-click the header *Join_1* under *Data Sources*.

b) In the context menu, choose *Add to Output*.

All five fields appear on the right side, under the *Output Columns* pane.

8. Change *PRODUCT_TEXT* to *PRODUCT_NAME* for both the label and also the name.

a) In the *Semantic* node, click inside the *PRODUCT_TEXT* field, under the *Name* column. Enter **`PRODUCT_NAME`**.

b) Click inside the field *PRODUCT_TEXT*, under the *Label* column. Enter **`PRODUCT_NAME`**.

9. Save and build the new calculation view. Then check the console to ensure that the calculation view has been successfully built.

a) Choose *Save*, or use the menu path *File* → *Save*.

b) Choose *Build* → *Build Selected Files*.

In the console, which appears at the bottom of the screen, you see a log. Ensure that you can see that the build completed successfully.

10. Preview the data for the calculation view *CVD_PRODUCT_##*. Check that you see the product group text displayed against all products (this proves that the join is working).

a) Right-click your calculation view that appears on the Development objects tree on the left of the screen. In the context menu, choose *Data Preview*.

b) Select the *Raw Data* tab.

c) Check that the *PRODUCT_GROUP_TEXT* column appears for each product and that the five output columns you chose display.

125

Create a Calculation View of Type Cube with Star Join

Exercise Objective

In this exercise, you learn how to build a calculation view of type **cube with star join** that combines the two dimension views that you created earlier with sales transactions to form a multi-dimensional model for flexible drill-down analysis.

Business Example

You have been asked to create a model that enables users to explore product sales data for US customers. It should be possible to present all sales figures, aggregated by any combination of customer and product attributes.

Before you start, preview the sales fact table to familiarize yourself with the data.

Refer to the course section, which covers the Database Explorer to learn how to access and browse the tables. Remember that we use synonyms to access the tables from the *TRAINING* schema in this exercise.

Database Schema: *TRAINING*.

Column Table: *SALES_DATA*.

Columns of interest: *CUSTOMER_ID*, *PRODUCT_ID*, *QUANTITY*, *QTY_UNIT*, *AMOUNT*, *CURRENCY*, and *SQL_DATE*.

Note:
In this exercise, replace ## with your own, two-digit student number.

Task 1: Create the basic calculation view of type: cube with star join

1. In the *exercises* folder, create a new calculation view with the following settings:

   Table 10: Calculation View Properties

Field	Value
Name	**`CVC_US_SALES_##`**
Label	**`US Product Sales`**
Data Category	*CUBE*
With Star Join	Select

2. Add an *aggregate* node to the flow and include the table *SALES_DATA*, from the *TRAINING* schema, to this node.

3. A star schema needs a central fact table. To create this, add the seven columns from the table, *Column Names*, to the output.

   Table 11: Column Names

Column Name
CUSTOMER_ID
PRODUCT_ID
QUANTITY
QTY_UNIT
AMOUNT
CURRENCY
SQL_DATE

4. Change the name of the aggregation node from *Aggregation_1* to *SALES_FACTS* to provide a better meaning for the fact data.
5. Connect the *aggregation* node to the *star join* node.
6. To create your star schema, add the *CVD_US_CUSTOMER_##* and *CVD_PRODUCT_##* dimension views to the *star join* node.
7. Join the *SALES_FACTS* data source to the two dimension views.
8. From the *star join* node, map the five columns, *QUANTITY*, *QTY_UNIT*, *AMOUNT*, *CURRENCY*, and *SQL_DATE* to the output.
9. Change *AMOUNT* to *SALES_REVENUE* and *SQL_DATE* to *SALES_DATE*.
10. In the *semantic* node, check that the correct column type has automatically been assigned to the columns *SALES_REVENUE* and *QUANTITY*.

    They should both be defined as a *Measure* and all the other columns defined as an *Attribute*.
11. Save and build the new calculation view. Then, check the console to ensure that the calculation view has been successfully built.
12. Preview the data for *CVC_US_SALES_##*. Check that you see the measures from the fact table are supported by attributes from both dimensions.
13. Use the *Analysis* tab to display the total sales quantity by product group.

Task 2: Enhance your calculation view with a calculated column

1. To indicate whether delivery is **free** or **chargeable** based on the customer location, generate a new column based on a column engine expression. Use the following information:

   Table 12: Calculated Column Data

Section	Field	Value
General	*Name*	`Delivery`

Section	Field	Value
General	*Label*	**`Delivery Charge`**
General	*Data Type*	**`VARCHAR`**
General	*Length*	**`10`**
Semantics	*Column Type*	**`Attribute`**
Expression	*Formula*	`If ("LOCATION"='Philadel-phia','FOC','CHARGEABLE')`

2. Save and build the calculation view. Then check the console log to ensure that the calculation view has been successfully built.
3. Preview the data for *CVC_US_SALES_##*. Check that you see the measures from the fact table supported by attributes from the dimensions.

128 Create a Calculation View of Type Cube with Star Join

Exercise Objective

In this exercise, you learn how to build a calculation view of type **cube with star join** that combines the two dimension views that you created earlier with sales transactions to form a multi-dimensional model for flexible drill-down analysis.

Business Example

You have been asked to create a model that enables users to explore product sales data for US customers. It should be possible to present all sales figures, aggregated by any combination of customer and product attributes.

Before you start, preview the sales fact table to familiarize yourself with the data.

Refer to the course section, which covers the Database Explorer to learn how to access and browse the tables. Remember that we use synonyms to access the tables from the *TRAINING* schema in this exercise.

Database Schema: *TRAINING*.

Column Table: *SALES_DATA*.

Columns of interest: *CUSTOMER_ID*, *PRODUCT_ID*, *QUANTITY*, *QTY_UNIT*, *AMOUNT*, *CURRENCY*, and *SQL_DATE*.

> Note:
> In this exercise, replace ## with your own, two-digit student number.

Task 1: Create the basic calculation view of type: cube with star join

1. In the *exercises* folder, create a new calculation view with the following settings:

Table 10: Calculation View Properties

Field	Value
Name	**`CVC_US_SALES_##`**
Label	**`US Product Sales`**
Data Category	*CUBE*
With Star Join	Select

a) In the content tree of the *Development* view, right-click the *exercises* folder and choose *New* → *Calculation View*.

b) Enter the properties provided in the table, *Calculation View Properties*.

c) Choose *Create*.

2. Add an *aggregate* node to the flow and include the table *SALES_DATA*, from the *TRAINING* schema, to this node.

a) From the node selection toolbar, click the *Create Aggregation* icon and then click in the empty space at the bottom of the data flow to add the new node.

The new aggregate node appears directly below the existing *star join* node.

b) Choose the *aggregate* node just created, and to open the data source selector, choose the  *Add Data Source*.

> Note:
> You might need to click the empty canvas and then navigate back to the node for the icon to appear.

c) In the *Search* field, enter **`sales`**.

d) Highlight the row where the table *SALES_DATA* appears.

e) Choose *Finish*.

3. A star schema needs a central fact table. To create this, add the seven columns from the table, *Column Names*, to the output.

Table 11: Column Names

Column Name
CUSTOMER_ID
PRODUCT_ID
QUANTITY
QTY_UNIT
AMOUNT
CURRENCY
SQL_DATE

a) Highlight the aggregation node and then choose the last icon in the list of action icons that appear to the right of the node.
The details panel expands and appears to the right of the canvas.

b) Ensure that the *Mapping* tab displays (it is the default).

c) Under the header *HA100 :: SALES_DATA*, a list of all available source fields appears. Right-click each field from the list provided in the table, and in the context menu, choose *Add to Output*.

The seven selected fields appear on the right under *Output Columns*.

4. Change the name of the aggregation node from *Aggregation_1* to *SALES_FACTS* to provide a better meaning for the fact data.
   a) Right-click the node *Aggregate_1*, and select *Rename* and overwrite the default name with the new label **SALES_FACTS**.
5. Connect the *aggregation* node to the *star join* node.
   a) Select the *aggregation* node so that the action symbols appear to the right.
   b) Drag a line between the arrow symbol and the circle at the bottom of the *star join* node above.
6. To create your star schema, add the *CVD_US_CUSTOMER_##* and *CVD_PRODUCT_##* dimension views to the *star join* node.
   a) Select the *star join* node then choose the  *Add Data Source* icon.
   b) In the *Search* field, enter **_##** (don't forget to replace ## with your assigned number).
   The two dimension calculation views that you created earlier appears.

   > Note:
   > Remember, you can also use the ready-made calculation views of type dimension that end with *_00* in case you didn't create one or both of your own from the previous exercise.

   c) Highlight the two rows where the calculation views *CVD_US_CUSTOMER_##* and *CVD_PRODUCT_##* appear.
   d) Choose *Finish*.
   They now appear in the canvas alongside the sales facts.
   e) Using drag and drop, arrange the views so that the sales facts are in the center of the canvas with the dimension views on opposite sides. This makes it easier to create a star schema in the next step.
7. Join the *SALES_FACTS* data source to the two dimension views.
   a) Draw a line from *PRODUCT_ID* in the *SALES_FACTS* data source to the *PRODUCT_ID* field in the *CVD_PRODUCT_##* dimension view.
   b) Draw a line from *CUSTOMER_ID* in the *SALES_FACTS* data source to the *CUSTOMER_ID* field in the *CVD_US_CUSTOMER_##* dimension view.
8. From the *star join* node, map the five columns, *QUANTITY*, *QTY_UNIT*, *AMOUNT*, *CURRENCY*, and *SQL_DATE* to the output.
   a) Select the *star join* node and then select the *Mapping* tab at the top of the screen.
   b) Under the header *SALES_FACTS*, you see a list of all seven available source fields. Right-click *QUANTITY* and in the context menu, choose *Add to Output*.
   The field appears on the right side, under the *Output Columns* pane.
   c) Repeat substep (b) for *QTY_UNIT*, *AMOUNT*, *CURRENCY*, and *SQL_DATE*.
9. Change *AMOUNT* to *SALES_REVENUE* and *SQL_DATE* to *SALES_DATE*.

a) Choose the *semantics* node. In the *Name* and *Label* columns for *AMOUNT*, enter **`SALES_REVENUE`**.

b) Choose the *semantics* node. In the *Name* and *Label* columns for *SQL_DATE*, enter **`SALES_DATE`**.

10. In the *semantic* node, check that the correct column type has automatically been assigned to the columns *SALES_REVENUE* and *QUANTITY*.

    They should both be defined as a *Measure* and all the other columns defined as an *Attribute*.

    a) Choose the *Semantics* node.

    b) Verify that the *Type* column displays the correct type for each row.

11. Save and build the new calculation view. Then, check the console to ensure that the calculation view has been successfully built.

    a) Choose *Save*, or use the menu path *File* → *Save*.

    b) Choose *Build* → *Build Selected Files*.

    c) In the console, which appears at the bottom of the screen, you should see a log. Ensure that you can see that the build completed successfully.

12. Preview the data for *CVC_US_SALES_##*. Check that you see the measures from the fact table are supported by attributes from both dimensions.

    a) Right-click your calculation view that appears on the Development tree on the left of the screen. In the context menu, choose *Data Preview*.

    b) Select the tab *Raw Data*.

    c) You should now see that the master data attributes (customer and product) appear alongside the sales amounts and quantities.

13. Use the *Analysis* tab to display the total sales quantity by product group.

    a) Still in the *Data Preview*, choose the *Analysis* tab.

    b) From the *Attributes*, drag *PRODUCT_GROUP_TEXT* to the *Label Axis*.

    c) From the *Measures*, drag *QUANTITY* to the *Value Axis*.

Task 2: Enhance your calculation view with a calculated column

1. To indicate whether delivery is **free** or **chargeable** based on the customer location, generate a new column based on a column engine expression. Use the following information:

   Table 12: Calculated Column Data

Section	Field	Value
General	*Name*	**`Delivery`**
General	*Label*	**`Delivery Charge`**
General	*Data Type*	**`VARCHAR`**
General	*Length*	**`10`**
Semantics	*Column Type*	**`Attribute`**

Section	Field	Value
Expression	*Formula*	`If ("LOCATION"='Philadel-phia','FOC','CHARGEABLE')`

a) In the *calculation view*, choose the *star join* node and choose the tab *Calculated Columns*.

b) Choose the **+** symbol and in the context menu, choose *Calculated Column*.

c) Using the first five entries from table, *Calculated Column Data*, define the basic settings of the calculated column.

d) Choose the *Expression Editor* button at the bottom of the screen.

e) Change the *Language* setting to *Column Engine*.

f) From the *Functions* tab, single-click the *If ()* function that you find when you expand the *Misc Functions* category.

g) In the *Expression Editor* pane, remove the *int* placeholder and leave the cursor exactly in its place.

h) In the *Elements* tab, expand *Calculation Views* and then expand *CVD_US_CUSTOMER_##*. Single-click the column *Location* so that it appears immediately after the first parenthesis and before the first comma.

i) From the *Operators* tab, choose the = symbol so that it appears immediately after *LOCATION*.

j) Immediately after the = symbol, type **`'Philadelphia'`** and be sure to include both single quotes to show that it is a string value.

k) After the first comma remove the *arg1* placeholder and replace it by typing **`'FOC'`** remembering to include both single quotes.

l) After the second comma remove the *arg2* placeholder and replace it by typing **`'CHARGEABLE'`** remembering to include both single quotes.

The full expression should be **`If("LOCATION"='Philadelphia','FOC','CHARGEABLE')`**.

m) To check for errors, choose the *Validate Syntax* button.

2. Save and build the calculation view. Then check the console log to ensure that the calculation view has been successfully built.

a) Choose *Save*, or use the menu path *File* → *Save*.

b) Choose *Build* → *Build Selected Files*.

c) In the console, which appears at the bottom of the screen, you should see a log. Ensure that you can see that the build completed successfully.

3. Preview the data for *CVC_US_SALES_##*. Check that you see the measures from the fact table supported by attributes from the dimensions.

a) Right-click your calculation view that appears on the Development tree on the left of the screen. In the context menu, choose *Data Preview*.

b) Select the *Raw Data* tab.

c) Check that the *Delivery* column now shows either 'FOC' or 'CHARGEABLE' for each product depending on whether the customer is based in Philadelphia or not.

Additional Considerations for Core Modeling

In the previous concept, we focused on calculation views. Calculation views are built using a graphical approach and no coding is required. However, sometimes the graphical approach does not provide all the functions and data flow logic you require for a complex calculation view. This is when you use functions. Functions are built using SQL Script and offer lots of flexibility to write simple or complex logic.

Note:
There are actually two type of functions — **scalar functions** (return a single value, such as current date) and **table functions** (return a tabular data set of many rows). In this section we focus on table functions.

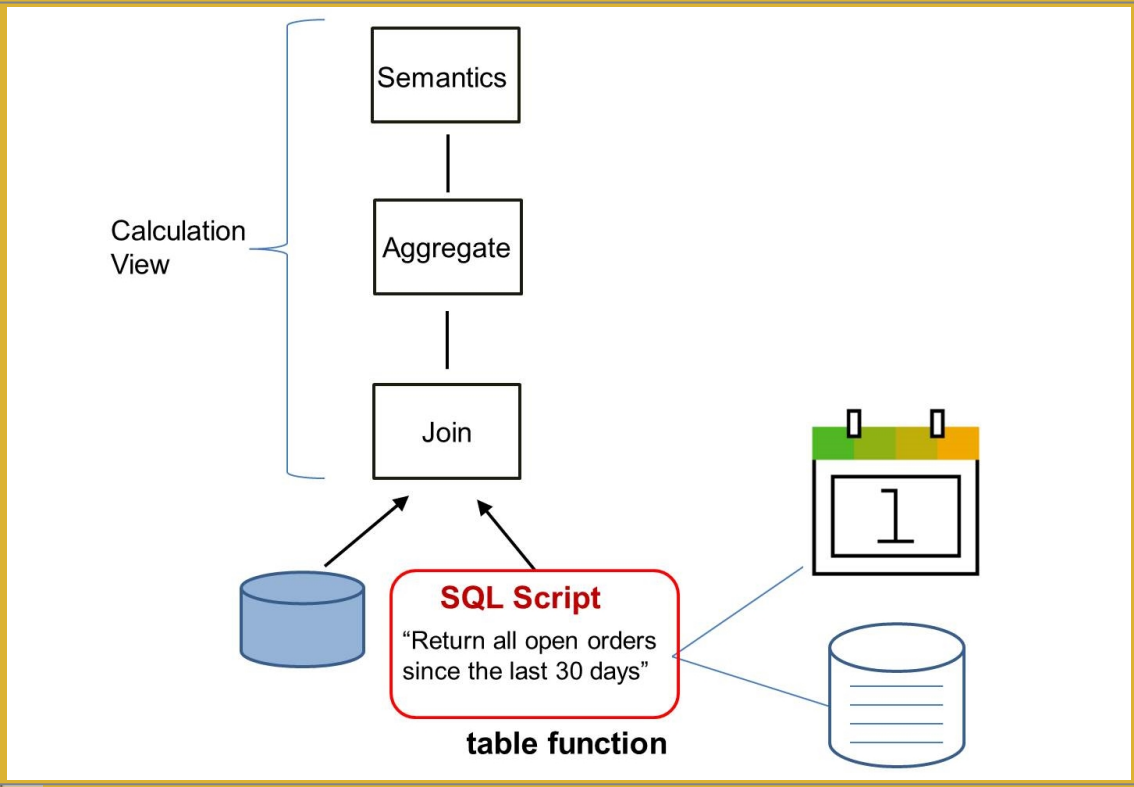


Figure 57: Table functions

In SAP HANA modeling, table functions are typically used to generate a tabular data set that is used as a data source in a calculation view. Table functions can be used in SQLScript in a FROM clause of a select statement (in other words wherever a standard table is used). A table function encapsulates the logic in a reusable form so that it can be used many times in different artifacts.

Table functions can accept one or more input parameters. Table functions are read-only; that is, they cannot be used to change data. Table functions produce exactly only one tabular output. Table functions can also call other functions.

Add SQL to Models Using Procedures

Procedures define reusable data processing logic that can be used to enhance a calculation view. Procedures are very similar to functions in that they are written in SQL Script and can have one or more inputs and they always have outputs. However, procedures can produce

multiple output data sets of different structures, whereas a table function can only return one tabular output data set. Procedures cannot be used as data sources to calculation views. They are used throughout the calculation view, especially where some processing logic is required, for example, to automatically derive values for an input parameter, or to return values used in analytic privileges.

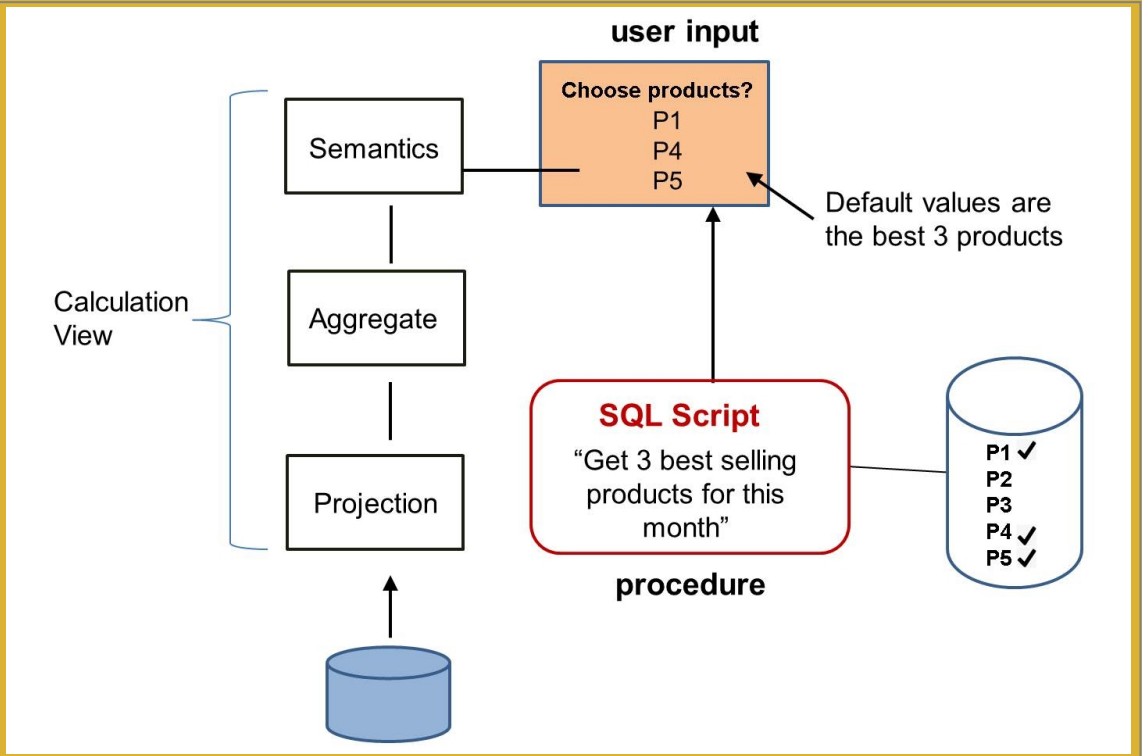


Figure 58: Procedures

A procedure can be called directly from an SQLScript, which means it can be called from a function or even another procedure.

Procedures used within modeling are mostly used as read only. In that case they are called stateless (or side-effect free), as they don't alter any data in the database. However, procedures can also be used to update, insert, and delete data. These procedures are called stateful, and they are not allowed when called from calculation views. Stateful procedures are more likely to be used by developers building applications.

In case you are wondering what happened to Decision tables from the first releases of HANA — they are still supported for SAP HANA 2.0 but only in the classic repository. You cannot build Decision tables in the new HDI framework and there is no support with SAP Web IDE. You have to manually re-create the rules logic in graphical calculation views (maybe using functions with SQL CASE logic). For writing complex, reusable business rules, SAP now recommends that customers use the new HANA Rules Framework (HRF). This offers significantly more depth and power and can accomplish everything that was possible with DT, and more. You write the business rules using an expression language instead of a fixed 'grid' as we have in DT. There is lots of online documentation on HRF to help you learn more. We didn't include HRF in this course as it is a little too deep and we don't want to overwhelm students. We also removed the Decision Tables topic from this HA100 course update, so as not to steer customers in the wrong direction. So, this is just for your own understanding, in case you are asked about it or want to include a short explanation of what happened to these classic modeling artifacts.

Security Considerations in Modeling

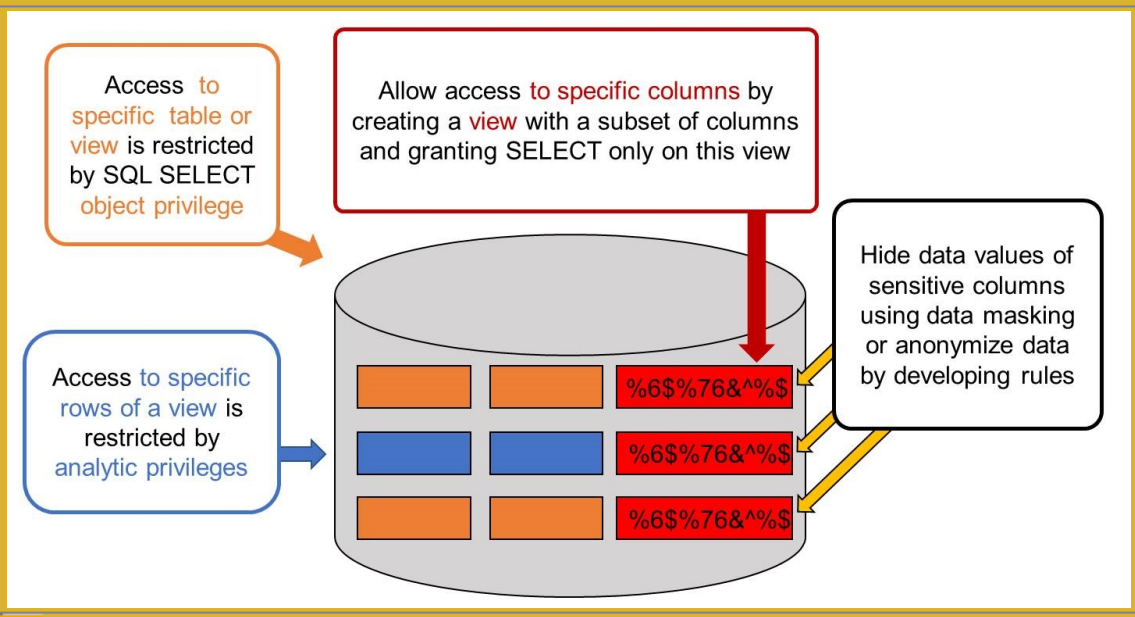


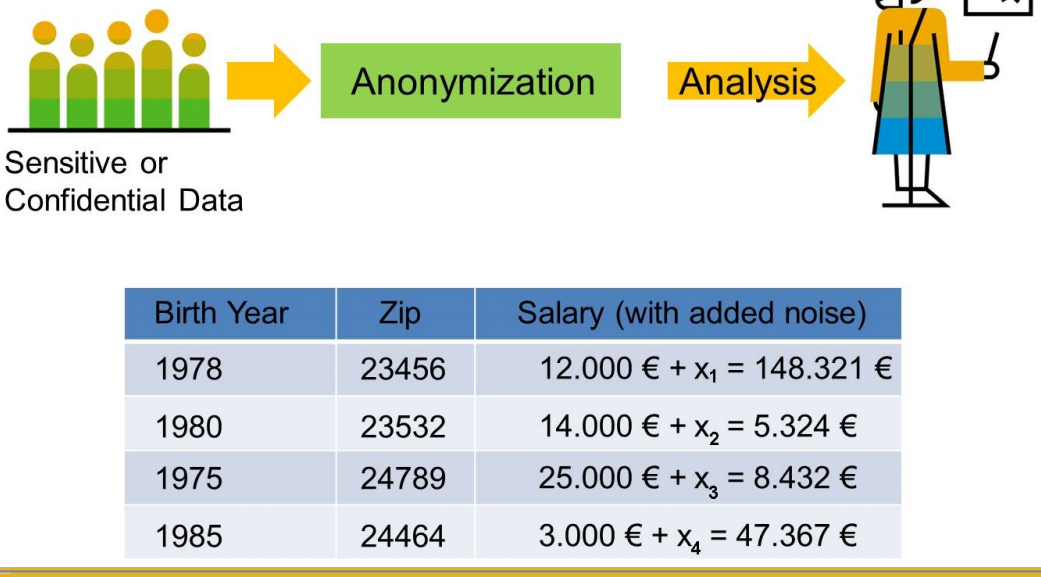
Figure 59: Security in modeling

Unless you grant access to users, they are not able to view any data from your calculation views. There are three levels of security, as follows:

1. The first level of access required is to the actual database objects. In this case we mean the calculation view and the source tables and functions that are included in the calculation view. This is achieved by granting a SELECT privilege on the calculation view. This privilege is granted to the user, or more likely, to the role the user is assigned to.
2. You need to have access at the row level. This is achieved by defining an **analytic privilege** and assigning it to the user, or the role to which the user is assigned. An analytic privilege is an object that is created in the SAP HANA Studio or Web IDE for SAP HANA. It describes for each calculation view the conditions for data access. The conditions can be simple such as User 1 can see company A data. The conditions can also be more complex such as User 1 can see company A and B data but only between 2017–2018. A user can have multiple analytic privileges. Analytic privilege logic can be written using simple rules such as these mentioned, or for more complex rules we can write SQL Script.
3. Then we come to the visibility of data values of each column. We can hide sensitive columns by applying a masking rule to each column. For example, instead of displaying a salary we only show £xxxxxx. We then only grant access to certain users who see the original data and not the mask.

Data Anonymization

Compliancy with data privacy laws is a big challenge for many organizations who would like to develop analytics over their data that includes people information, but they are aware that new rules protect the individuals identity and how their data is used. This means that often organizations are reluctant to start new projects for fear that they might be at risk of breaking data privacy laws.



Birth Year	Zip	Salary (with added noise)
1978	23456	12.000 € + x_1 = 148.321 €
1980	23532	14.000 € + x_2 = 5.324 €
1975	24789	25.000 € + x_3 = 8.432 €
1985	24464	3.000 € + x_4 = 47.367 €

Figure 60: Data Anonymization

Since SAP HANA 2.0 SPS03 we can build sophisticated data anonymization rules into our data models that protect a person's privacy. We can apply rules to sensitive data so that it is not possible to identify individuals from that data. For example, instead of displaying total sickness days taken by birth date (where it might be possible to identify someone by their birthday), we might want to avoid using exact dates and instead display sickness days by birth year or even ranges of years. This method hides people in a crowd.

Another anonymization method is to add noise to data to make it impossible to derive any meaningful information at an individual personal level. But when aggregated, the data makes sense. For example add or subtract values to an individual's salary, but the overall additions and subtractions net to zero so what remains are the true salary values.

Note:
For more information about security in SAP HANA, refer to the detailed course, HA240.

LESSON SUMMARY

You should now be able to:

- Understand SAP HANA core modeling

Understanding SAP Supplied Data Models

139

LESSON OVERVIEW
In this lesson we will describe the two types of SAP HANA virtual data models delivered by SAP.

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Describe SAP HANA Live
- Describe SAP S/4HANA Embedded Analytics

SAP HANA Live

SAP HANA Live is a complete data model built and supplied by SAP. SAP HANA Live is built with calculation views. These calculation views expose live transactional and master data from the tables of SAP Business Suite applications. This includes SAP ERP, SAP CRM, SAP SCM, and so on. SAP HANA Live is not relevant for S/4HANA.

SAP HANA Live Architecture

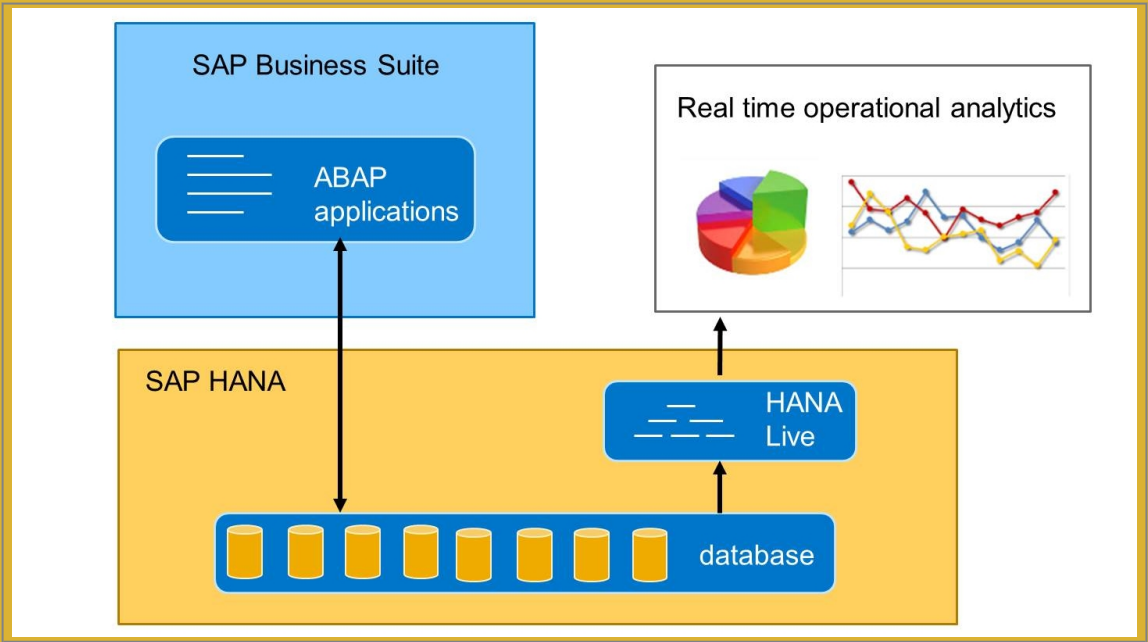


Figure 61: SAP HANA Live Architecture

Due to the high degree of normalization of the database schemas found in SAP Business Suite applications, tables can appear complex, fragmented, and are often difficult to consume without a thorough understanding of each schema. SAP HANA Live provides ready-to-use business views of the SAP Business Suite data to enable a fast start with live operational

reporting. A key benefit of SAP HANA Live is that developers can easily consume live business information from SAP Business Suite without being concerned about the underlying table complexities. The calculation views are created and maintained by SAP using standard SAP HANA modeling tools, so that they can be viewed and modified easily by customers using well known tools.

The key use case for SAP HANA Live is **real-time, operational analytics** on SAP Business Suite. SAP HANA Live is a read-only data model, so it is of no use for applications that need to write data back to the database. SAP HANA Live can be consumed immediately by SAP Business Objects reporting tools and, in fact, any clients that is able to consume standard SAP calculation views.

Key Aspects of SAP HANA Live

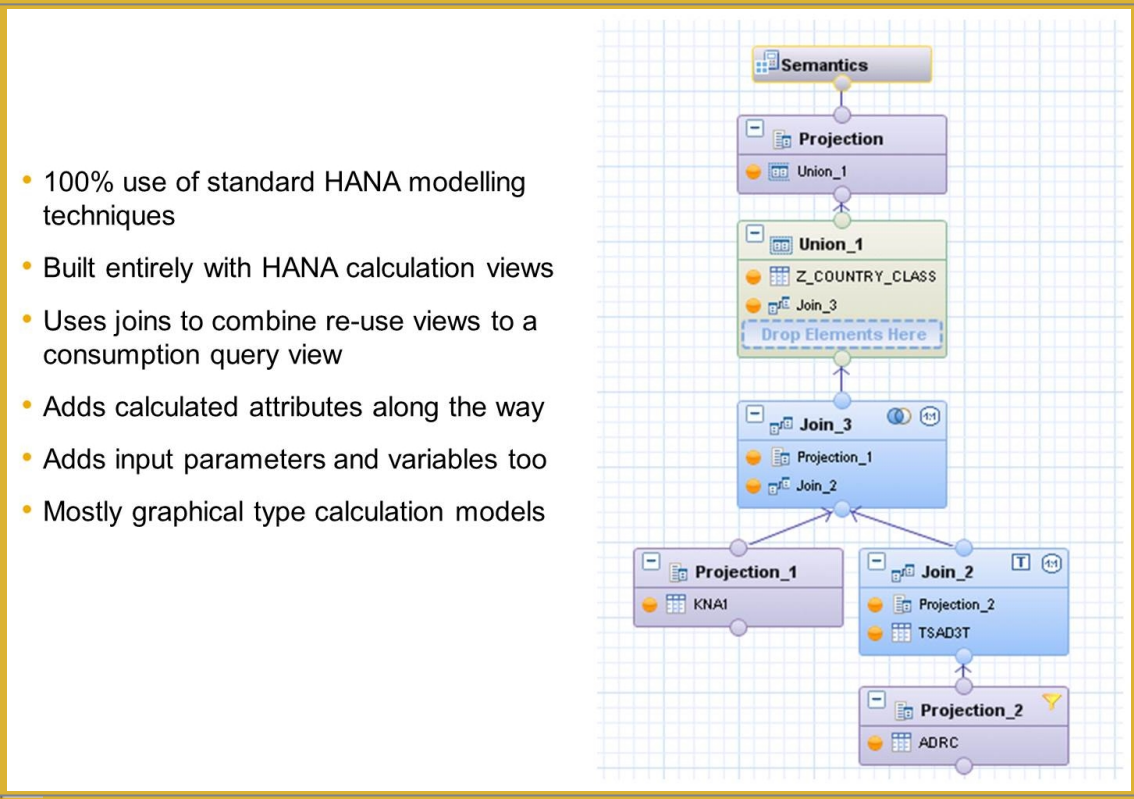


Figure 62: Key Aspects of SAP HANA Live

SAP HANA Live models contain all the necessary joins, filters, aggregations, and transformations to transform the data in the raw Business Suite tables into meaningful information. They are compiled into column views that are technically no different from the column views created from customer calculation views.

SAP HANA Live views do not aggregate data. They expose the data at the most granular level. However, the reporting tools are able to aggregate the data as required. They can be consumed by any SAP Business Objects reporting tool or any client that is able to consume a standard SAP HANA calculation view.

Most SAP HANA Live tools are integrated into SAP HANA Studio. However, some SAP HANA Live tools such as SAP Live Browser, are built using SAPUI5 and are not part of Studio.

SAP HANA Live is optional and can be freely downloaded and installed into SAP HANA. It is not a licensed product.

SAP HANA Live has its own release cycle. So, check the versions available and evaluate which version works with your SAP Business Suite and SAP HANA solution combination.

SAP HANA Live is based on the classic repository based development approach and not the new HDI container based development approach that is often associated with SAP HANA 2.0. There are no plans to migrate SAP HANA Live to this approach and it will remain in the classic repository / non-container approach.

Side by Side or Integrated Approach

A key decision that customers must make is to decide whether to implement SAP HANA Live as an integrated scenario or as a side-by-side deployment.

Side-by-side deployment is used when the SAP Business Suite application does not run on SAP HANA and a separate SAP HANA system is deployed to host SAP HANA Live. Although this approach means that the SAP Business Suite is able to continue to run on a non-HANA database, it means that a data movement tool would need to be implemented to copy data from the tables in the source application to the duplicated tables in SAP HANA. SAP Landscape Transformation Replication Server (SLT) is recommended for this, but other data provisioning tools can also be used. The key objective of the data movement tool is to ensure that operational tables are replicated in real time to SAP HANA.

The alternative is to enter all tables manually. Note that with a side car deployment, the data exposed by the SAP HANA Live view is only as up-to-date as the last replication of data from the source tables. So, if you set your replication to ten minute intervals, you do not really have a 'live' view of your source data.

Ideally, all operational data should be replicated in real-time to provide up-to-date information, hence the name SAP HANA Live.

The integrated deployment scenario is used when the SAP Business Suite application is already running on SAP HANA Live. SAP HANA Live is installed right inside the SAP HANA platform that is used to power the Business Suite application. The benefit of this approach is simplicity as the tables used by the SAP Business Suite applications are the same tables on which SAP HANA Live calculation views are based. No data movement takes place. This means that the SAP HANA Live views always reflect real-time information with no copying of data needed.

SAP S/4HANA Embedded Analytics

Embedded Analytics is a core component delivered with SAP S/4HANA and provides tools and a complete data model that exposes **live, operational data from SAP S/4HANA**. You can think of this as broadly the equivalent of SAP HANA Live (which is used with SAP Business Suite).

Like SAP HANA Live, Embedded Analytics data model provides a large number of ready-to-use views of transactional and master data, but this time the views are based on S/4HANA tables. S/4HANA tables and Business Suite tables are designed differently and this is why a new data model was needed for S/4HANA and why we couldn't simply re-use SAP HANA Live. The views supplied with Embedded Analytics are built using a technology called Core Data Services (CDS). CDS is built with a scripting language, rather than the graphical approach that we use for calculation views. CDS allows us to define the data definition and also the consumption model. In simple terms this means you can use CDS to define tables (data definition) and also the views (consumption model) over the tables.

For Embedded Analytics, CDS is used only to define the views, not the data definition. As well as the technical definition of the data model such as column length, types, joins etc, CDS allows you to fully describe the business semantics right in the model. These semantics can be consumed by various applications and provide additional information about the data. For

example, in the data model you can describe to the consuming client the correct way to aggregate a specific measure, or to identify certain fields as belonging to a customer address so that an application can simply refer to 'address' and not each individual column of the address. This is very similar to how we define the analytical rules in calculation views, except with CDS, we use code.

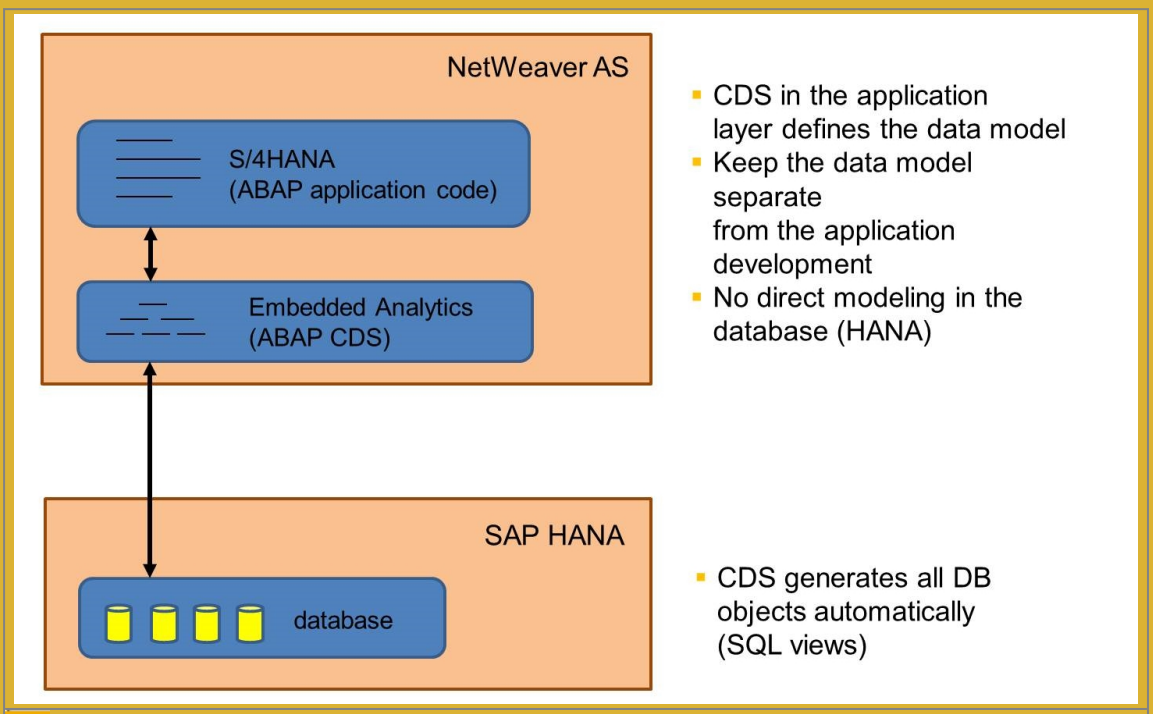


Figure 63: ABAP CDS

There are two types of CDS: HANA CDS and ABAP CDS. Embedded Analytics is based on ABAP CDS. This type of CDS is built in the ABAP layer of the stack, whereas HANA CDS is built in the database layer (in HANA). Both types of CDS broadly provide the same solution but ABAP CDS is closely coupled to the S/4HANA application development lifecycle, which is why it was chosen over HANA CDS, which is coupled to database development.

Comparing the Embedded Analytics data model using CDS with SAP HANA Live

First of all, CDS has more use cases than SAP HANA Live. SAP HANA Live is used for BI cases only, whereas CDS we have BI but also more cases such as search applications and planning applications.

Another key difference between SAP HANA Live and SAP HANA CDS is the method used to develop them. SAP HANA Live was created by SAP using the graphical SAP HANA modeling tools, which is more comfortable for most BI developers. ABAP CDS is developed using a code editor, similar to SQL and this is not so comfortable for BI developers and perhaps more aligned to application coders. Having said that, many BI developers are already starting to learn CDS just as they did with SQL.

Note:

SAP has a course on Embedded Analytics: S4H400 and a deep dive course on ABAP CDS: S4D430.

LESSON SUMMARY

You should now be able to:

- Describe SAP HANA Live
- Describe SAP S/4HANA Embedded Analytics

Unit 3
Lesson 3

Advanced Modeling with SAP HANA

145

LESSON OVERVIEW
In this lesson we will cover text processing with SAP HANA.

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Advanced Modeling with SAP HANA

Text Analytics with SAP HANA

Introducing SAP HANA Text Analytics

The vast majority of digital data is unstructured and a significant portion of that unstructured data is textual data. SAP HANA provides powerful text processing capabilities that can generate high value results from text. SAP HANA Text Analytics can be performed over many languages (32 at the current time) and on many text friendly data types. We can process text from many types of documents including PDF, PPT, DOC, HTML, XML, Outlook. This also includes binary documents.

SAP HANA Text Analytics can be broken up in to three major analytic capabilities:

- Full text search
- Text Analysis
- Text Mining

Full Text Search

Let's take a brief look at text search. It is called Full Text Search as opposed to simply Text Search because most text searches operate on individuals words. But SAP HANA Full Text Search works across entire phrases. This provides very powerful search possibilities.

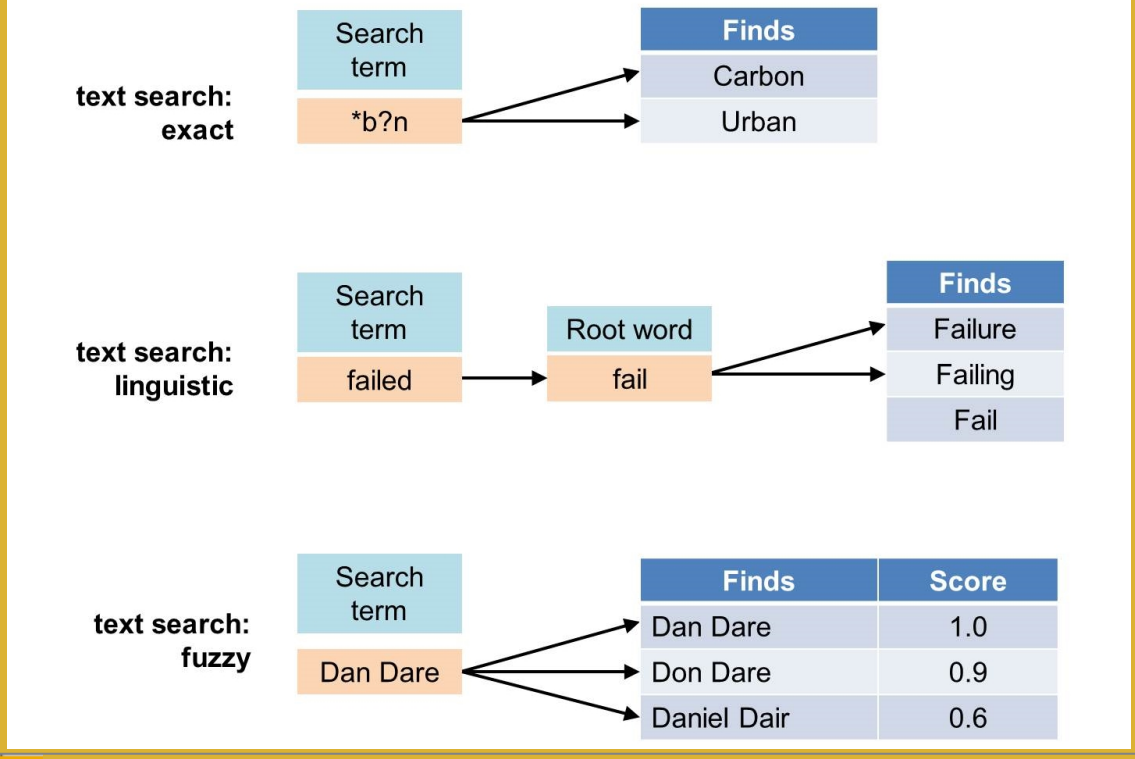


Figure 64: Text Search

With SAP HANA Full Text Search you can perform different types of text search:

- Exact search
  — search using wild cards pattern matching (for example,'*bo?' finds 'labor' and 'robot')
- Linguistic search
  — search using a linguistic analysis (for example, 'speaking' finds 'speak' and 'spoken')
- Fuzzy search
  — fault tolerant search (for example, 'ACME' finds 'AMCE')

Text Analysis

Text Analysis allows the extraction of structured information from unstructured information. An example of this is linguistic markup, which entails identifying the various parts of a speech (verbs, nouns, adjectives, and so on). It also allows you to identify entities (locations, persons, and dates) in an unstructured text. Once the linguistics has been extracted from the text we can perform powerful analytics on the results. For example, we can identify the subject, urgency, sentiment, and more.

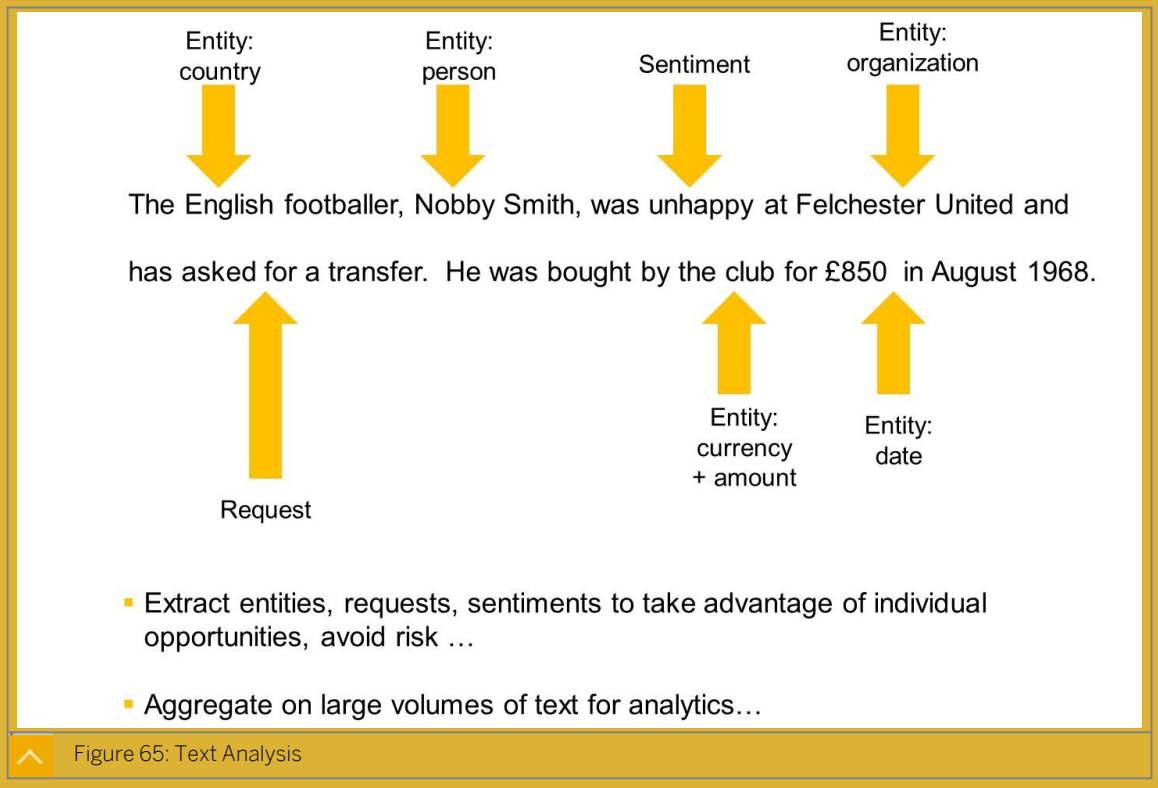


Figure 65: Text Analysis

The results of a Text Analysis are stored in a table and can therefore be easily consumed through all supported SAP HANA scenarios.

SAP provides ready-to-go reference dictionaries of entities in multiple-languages that can help identify interesting words that refer to organizations, dates, countries, and so on. SAP also supply a populated dictionary called “Voice of Customer”. This stores known words and expressions that customers use to express their feelings about products and services. It can be used in sentiment analysis to help organizations quickly respond to customer feedback on social media and other places where textual feedback is found. Customers can easily create their own custom dictionaries to include words and expressions used in their organizations.

Note:
You can launch a demo application to work with Text Analysis and Text Mining. The URL is found under the course folder *HA100 → URLs → Text Analysis and Mining Demo*. You can enter you own text or press *Browse Sample Inputs* to choose a ready-made document. Then select your choice of Configuration. As a suggestion, choose *Voice of Customer* configuration and then imagine you are writing a comment relating to how you feel about a product and what action you would like taken.

Text Mining

In contrast to text analysis, which operates on the words inside a document, text mining operates at the document level. In other words, text mining is used when the whole document is of interest and not just an analysis of some words or phrases within it.

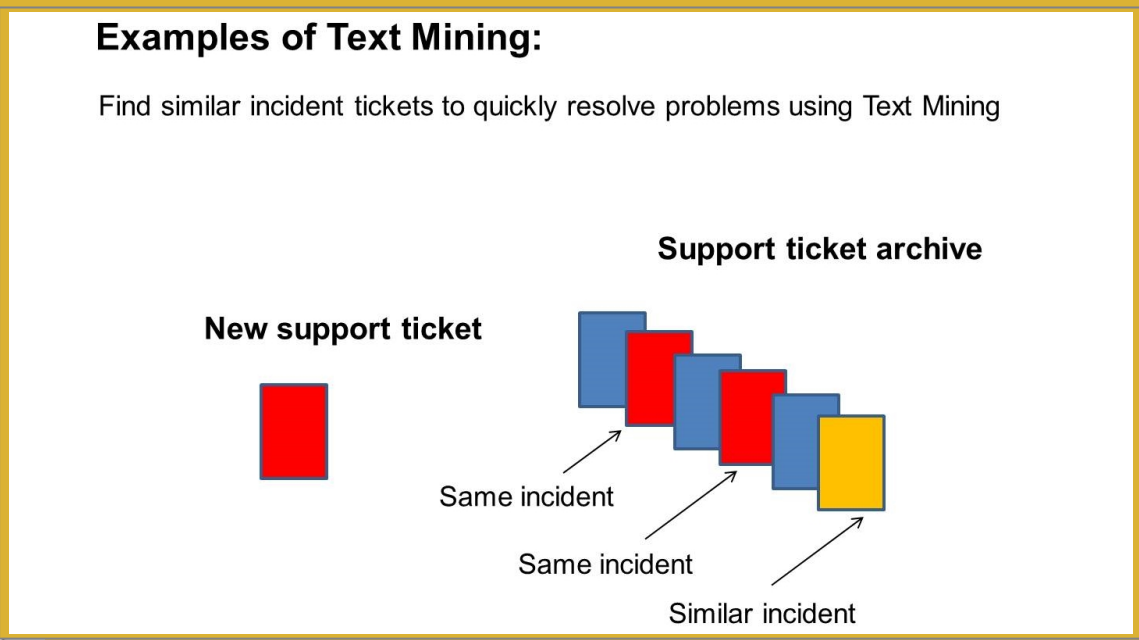


Figure 66: Text Mining

Text mining works with groups of documents to compare and classify using statistical analysis and not linguistic analysis. A key use case for text mining is to automatically classify streams of documents, such as newspaper articles, blogs, or support incidents. Or to locate similar documents, such as medical journal that discuss common topics. Each document is pre-processed and the strong subjects that they contain are noted and can be used to compare with other documents.

SAP HANA Spatial

By combining traditional business data with spatial data, you can build innovative applications that can provide deeper insight to unlock opportunities or reduce risk.

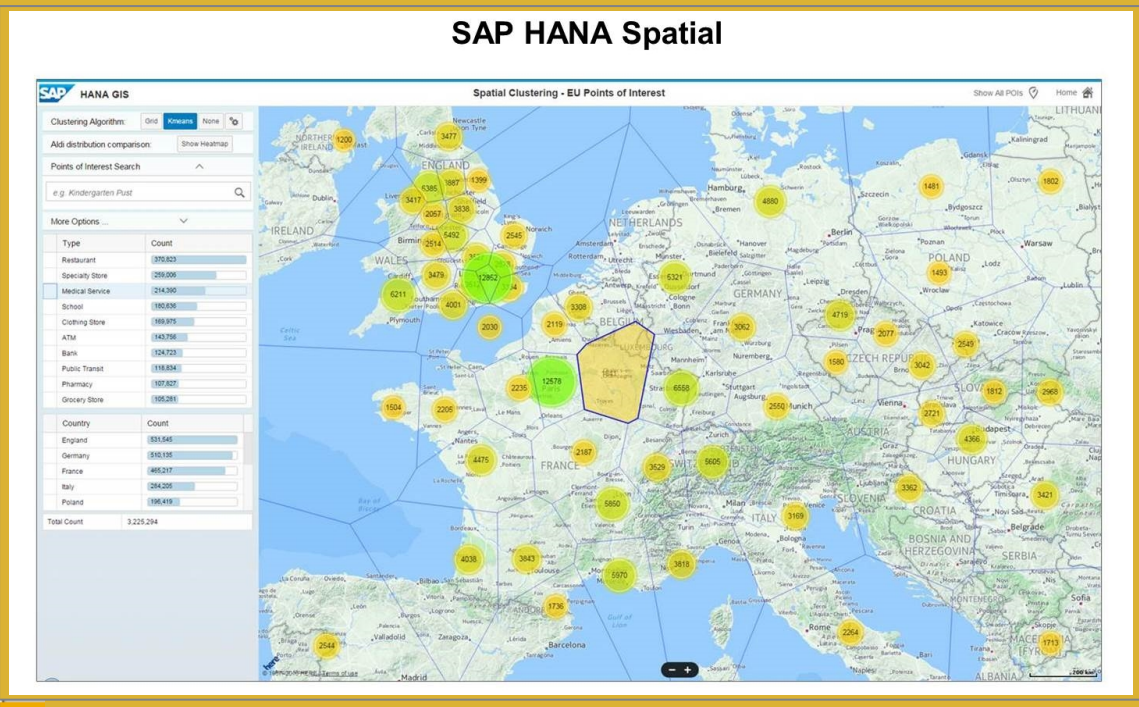


Figure 67: SAP HANA Spatial

Many organizations already rely on spatial data processing and use specialist applications alongside, but separate from, their business process applications. For example, a gas storage tank is leaking and an emergency repair order is raised in the SAP ERP system by a coordinator. Next, the coordinator uses a separate geo-based application and manually enters the asset number of the tank. The tank shows up on a map, and the coordinator then uses another application to identify the nearest engineer, who is then dispatched.

Missed Opportunities with Information Silos

It would be more efficient if, at the time of generating the repair order, the ERP application was able to locate the tank. It could then identify and dispatch the nearest qualified engineer who has enough working hours left to complete the repair, and provide useful geographic information to the engineer to describe how to best reach the tank quickly. The application could then provide information about other equipment in the close vicinity that is due an inspection soon. This prevents having to make separate visits. This scenario could be possible if the core business processes were integrated with spatial data and analysis.

Beyond business applications, there are more exciting use cases for spatial analysis in the sports environment. SAP have developed a series of applications that provide deep analysis of player performance. For example, in golf, by adding a sensor to the ball and pin, we can create a graphical history to illustrate the improvements in accuracy of the shot. These types of applications are already in use by major sports organizations around the world.

There are many applications that could be dramatically enhanced with the integration of spatial data.

SAP Spatial provides new data types for storing geometrical data such as points, lines, and polygons. These can be used to represent precise locations, roads, and regions. SAP HANA Spatial uses open standards and so can easily be integrated with well-known, leading geospatial providers such as ESRI, OGC, OpenStreepMap, GoogleMap.

As well as storing spatial data, SAP HANA also provides spatial query functions that can easily be included in SQL Script. Here are some examples of the functions:

- Within — which customers are in my region?
- Distance — what is the longest distance a high-value customer has to travel to reach my sales outlet?
- Crosses — where does truck route A cross truck route B?

SAP HANA Spatial also provides algorithms that can determine clusters. This helps an organization to locate precise locations that might be lucrative based on income data and other interesting attributes associated with consumers.

SAP HANA Spatial

By combining traditional business data with spatial data, you can build innovative applications that can provide deeper insight to unlock opportunities or reduce risk.

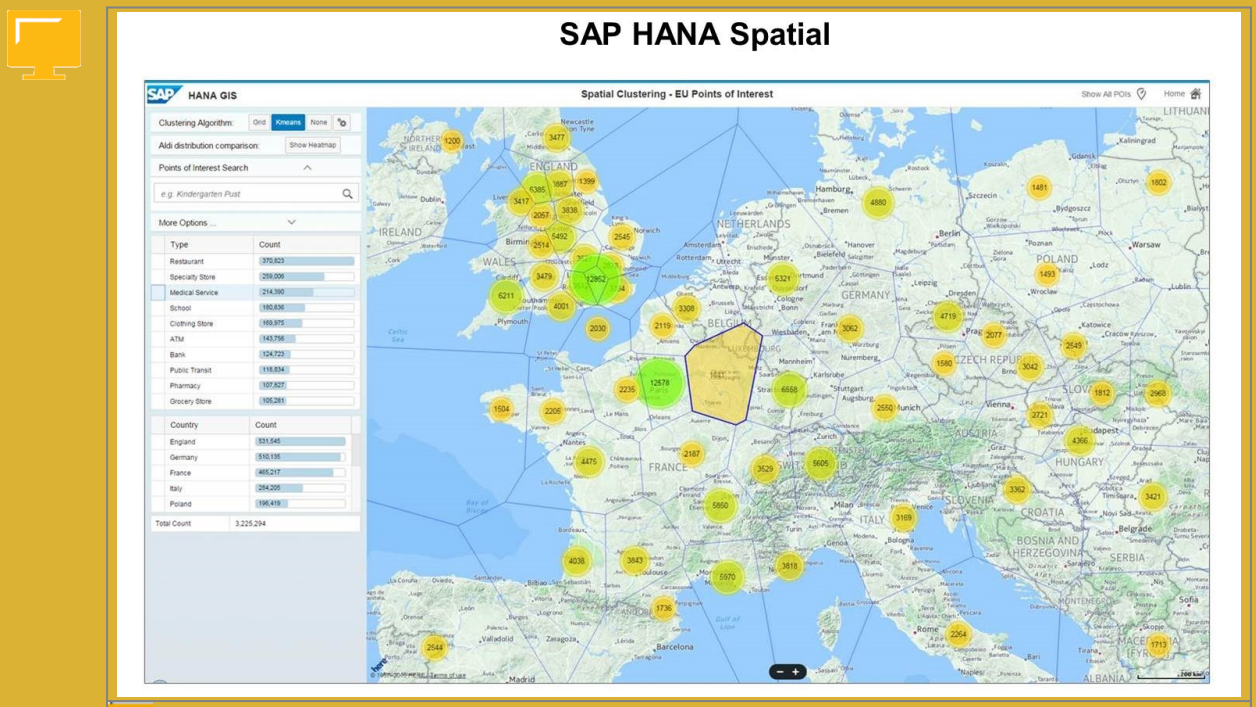


Figure 68: SAP HANA Spatial

Many organizations already rely on spatial data processing and use specialist applications alongside, but separate from, their business process applications. For example, a gas storage tank is leaking and an emergency repair order is raised in the SAP ERP system by a coordinator. Next, the coordinator uses a separate geo-based application and manually enters the asset number of the tank. The tank shows up on a map, and the coordinator then uses another application to identify the nearest engineer, who is then dispatched.

Missed Opportunities with Information Silos

It would be more efficient if, at the time of generating the repair order, the ERP application was able to locate the tank. It could then identify and dispatch the nearest qualified engineer who has enough working hours left to complete the repair, and provide useful geographic information to the engineer to describe how to best reach the tank quickly. The application could then provide information about other equipment in the close vicinity that is due an inspection soon. This prevents having to make separate visits. This scenario could be possible if the core business processes were integrated with spatial data and analysis.

Beyond business applications, there are more exciting use cases for spatial analysis in the sports environment. SAP have developed a series of applications that provide deep analysis of player performance. For example, in golf, by adding a sensor to the ball and pin, we can create a graphical history to illustrate the improvements in accuracy of the shot. These types of applications are already in use by major sports organizations around the world.

There are many applications that could be dramatically enhanced with the integration of spatial data.

SAP Spatial provides new data types for storing geometrical data such as points, lines, and polygons. These can be used to represent precise locations, roads, and regions. SAP HANA Spatial uses open standards and so can easily be integrated with well-known, leading geospatial providers such as ESRI, OGC, OpenStreepMap, GoogleMap.

As well as storing spatial data, SAP HANA also provides spatial query functions that can easily be included in SQL Script. Here are some examples of the functions:

- **Within** — which customers are in my region?
- **Distance** — what is the longest distance a high-value customer has to travel to reach my sales outlet?
- **Crosses** — where does truck route A cross truck route B?

SAP HANA Spatial also provides algorithms that can determine clusters. This helps an organization to locate precise locations that might be lucrative based on income data and other interesting attributes associated with consumers.

Predictive Analysis

With SAP HANA, you can develop predictive models using a variety of approaches. These approaches include using:

- Predictive Analysis Library (PAL)
- Automated Predictive Library (APL)
- Extended Machine Learning (EML)
- R (algorithm library)

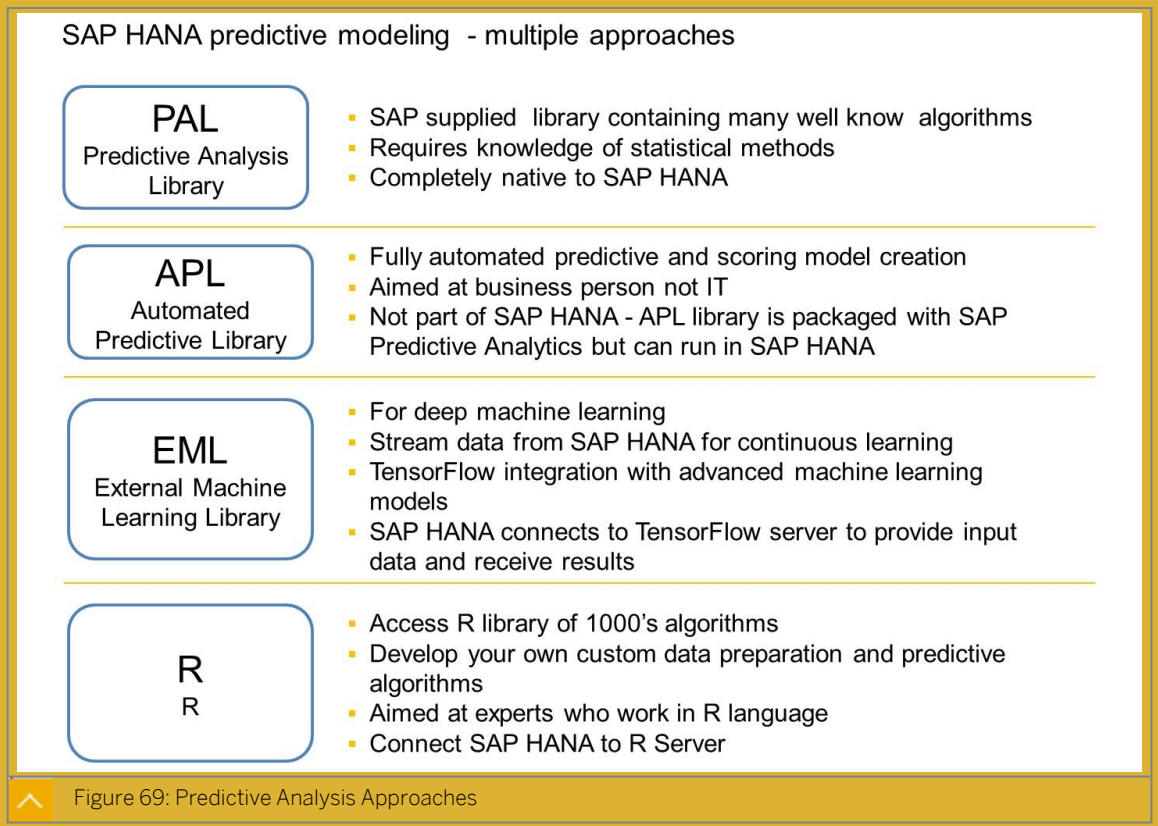


Figure 69: Predictive Analysis Approaches

SAP HANA Predictive Analysis Library (PAL)

SAP HANA Predictive Analysis Library (PAL) contains over 90 algorithms that can be used to develop predictive and machine learning models. Some of these algorithms are used for data mining pre-processing tasks such as:

- Sampling — select a few records from large data sets (for example, we need 1000 people from each country)

- Binnig — grouping records into basic categories (for example, age ranges)
- Partitioning — creating sets of data for training, testing, and validation used to train models and check their predictive accuracy

The majority of the algorithms are used for scoring or predictive modeling. There are many algorithms provided for all major data mining categories including:

- Association
- Classification
- Clustering
- Regression
- Time Series
- Neural Networks

PAL algorithms can be called directly from procedures in SQL Script or they can be integrated into an SAP HANA flowgraph which is built using a graphical editor in the Web IDE. A flowgraph defines the data inputs, data processing, and outputs and parameters used in the predictive model. Using PAL requires knowledge of statistical methods and data mining techniques. This is because the choice of which algorithm to use must be made by the developer. So it is important that the developer initially understands the differences between the algorithms used for data preparation, scoring, and prediction. But they must also know how to fine-tune the algorithms to reach to desired outcome. For example, a developer would need to decide when to use the time series algorithm for double exponential smoothing versus triple exponential smoothing and then how to adjust the parameters to consider trend or to eliminate outliers. Developers who work with SAP HANA PAL are typically already working in predictive analysis projects or have a reasonable understanding of the topic.

Automated Predictive Library (APL)

The Automated Predictive Library (APL) is not shipped with SAP HANA but belongs to the product **SAP Predictive Analytics**. For customers who use this product they can run their models in SAP HANA and that is why it is mentioned here. APL is aimed at business users who do not have (or desire to have) detailed knowledge of the algorithms and the maths behind the models. The selection of algorithms for data preparations, scoring, and predictions is completely automated (hence the name). All the business user has to do is to provide the data and APL finds the best data preparation and predictive models.

External Machine Learning Library (EML)

SAP HANA provides access to external libraries of machine learning models. Google TensorFlow was the first library to integrate with SAP HANA and today provides access to a large library of deep machine learning models. Examples of deep machine learning could include voice recognition, handwriting recognition, image recognition and more. The machine learning models are created in theTensorFlow framework and a TensorFlow client is installed in SAP HANA to create the connection to the TensorFlow server. SAP HANA sends the source data, such as an image, to the TensorFlow server where it is processed (either in training mode, testing, or production mode) and the result is passed back to SAP HANA.

Google Tensorflow has become a very popular and a vastly growing framework open sourced by Google for machine learning data flows and its recent integration with SAP HANA is essential for the support of the growing applications for IoT.

R

R is an open source programming language used to develop statistical models. The R library contains thousands of freely accessible algorithms that can be used or adapted to provide custom predictive and data preparation capabilities. SAP HANA connects to an R server where the data is processed and the results are passed back to SAP HANA. R language is becoming increasingly popular among Data Scientists as the standardized language for statistical computing.

Graph Modeling

Graphs are used to model data that is best represented using a network. Examples include supply chain networks, transportation networks, utility networks, and social networks. The basic idea behind graph modeling is that it allows a modeler to define a series of entities (nodes) and link them with lines (edges) to create a network. This network represents how each node relates to all other nodes.

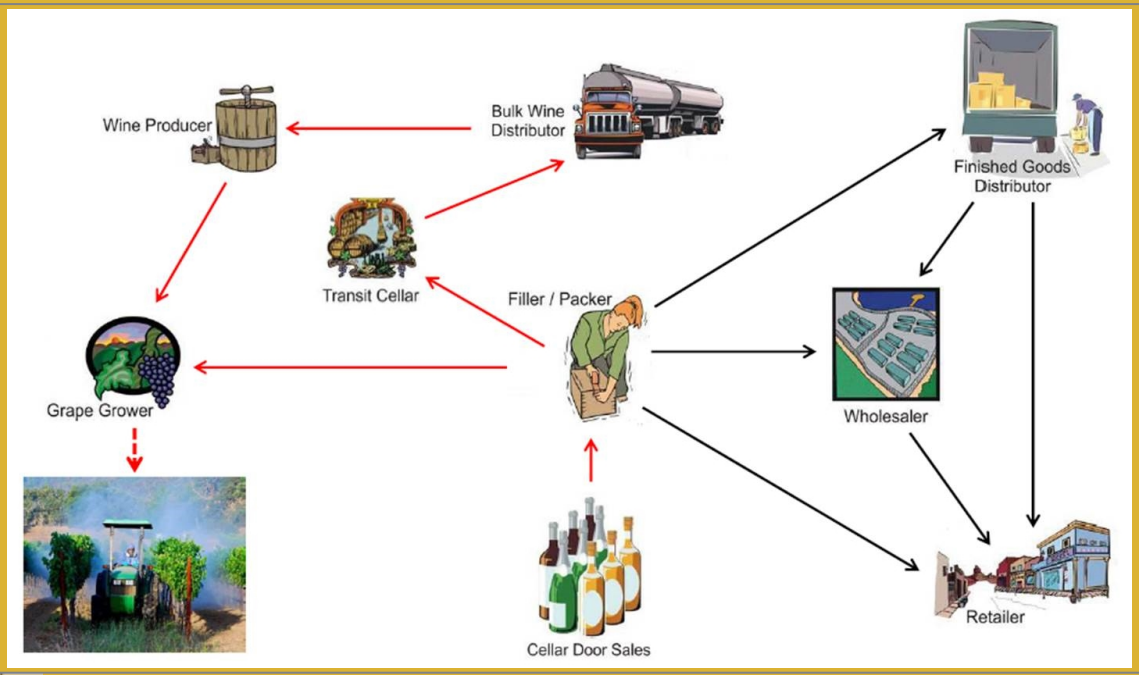


Figure 70: Graph Model: Example

Graph models can also indicate flow direction between entities and also any number of attributes can be added to nodes or the lines that connect them. This means that additional meaning can be added to the network and queries can be executed to ask questions relating to the network.

Imagine a complex supply chain mapped using a graph, where all manufacturers, suppliers, distributors, customers, and consumers are represented with information stored along the connections. The benefit to this form of modeling is that it makes it easy to develop applications that can traverse huge graphs at speed. As a result you can ask questions such as the following:

- How many hours has the product traveled between two specified points in the network?
- Where are all the possible points of origin of this product?
- Describe the entire journey of a product by listing all of the stop off points in the path

- What is the shortest path between point A and point B?

Graph processing allows you to discover hidden patterns and relationships in your network data, and all in real time.

Graph Model: Example

There are many examples of where SAP HANA Graph could be used, including the following:

- Medical

  Create a network of patients, conditions, treatments, and outcomes for reuse in diagnosis and planning treatments of other patients.

- Social Network

  Using popular social media portals, find your customers and their friends, friends of friends, and likes or dislikes to create marketing opportunities.

- Text Analysis

  An SAP HANA Text Analysis stores the results in a flat table structure. But sometimes the individual words in text can have multiple relationships to the other words in the same text. Storing the results of a text analysis in a graph provides the optimal model for querying the relationship between words. Without a graph model you would have to create a separate row in the table to represent the relationship from one word to all other words.

It is possible to use standard SQL tables with standard data definitions and query code to create and process a similar model. However, it would be extremely complex to define such a model with SQL and also to query the model. Also, processing times could be challenging. SAP HANA Graph provides tools for graph definition and additional language for graph querying to ensure that model development is more natural and simplified. It also guarantees that the processing is flexible, and of course, optimized for in-memory processing using a dedicated in-memory graph engine.

Series Data

When you collect data at a measurable interval such as time, the data is called **series data**. Analysis of series data allows you to draw meaningful conclusions and predictions from the patterns and trends present in the values.

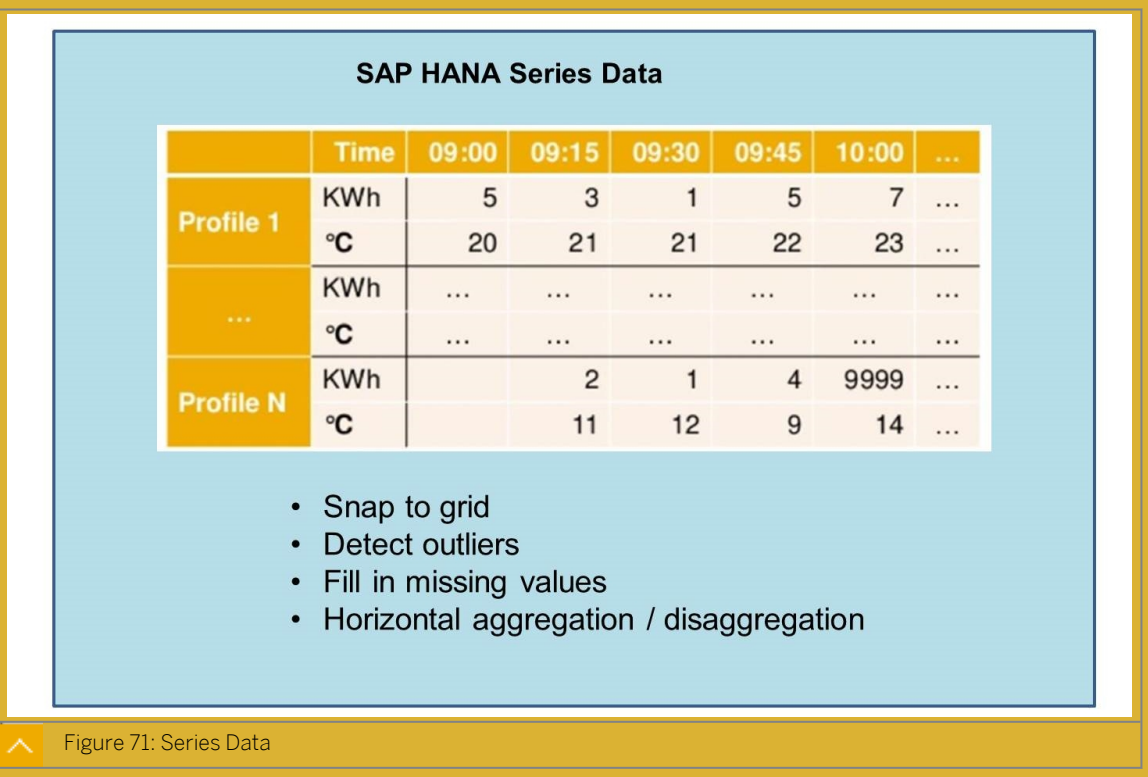


Figure 71: Series Data

A good example of series data that most people can relate to is when we consider energy metering. Energy companies are installing smart meters in households and businesses and are then able to collect energy consumption data at regular, and more frequent intervals that even before. SAP HANA Series Data provides efficient storage of this data using compression and also analytical capabilities on this data.

Some examples of the analytical capabilities of SAP HANA Series Data:

- Horizontal disaggregation — display meter readings that were collected at a course grain and disaggregate to a finer grain (for example, collect data at hourly readings and display are five minute intervals)
- Fill in missing values — if reading are missing, add them in
- Detect abnormal readings
- Adjust reading to nearest hour — when reading are not collected exactly on the hour, snap them to either the previous hour or the next hour depending on your rules.

There are many use cases for SAP HANA Series Data. Think of any scenario where data is collected at regular intervals, not necessarily relating to time intervals, maybe relating to an recurring event. The key word is **series**.

LESSON SUMMARY

You should now be able to:

- Advanced Modeling with SAP HANA

Learning Assessment

157

1. What is the role of core modeling in SAP HANA?

   Choose the correct answers.

   - [X] A To push data-intensive processing away from the application and to the database
   - [X] B To develop re-useable data processing logic in the database
   - [X] C To simplify application code

2. What are valid types of calculation view?

   Choose the correct answers.

   - [X] A Dimension
   - [X] B Cube
   - [X] C Cube with star schema
   - [X] D Dimension with star schema

3. If I need to create multiple output data sets using SQL Script. Which HANA object do I use?

   Choose the correct answer.

   - [X] A Procedure
   - [X] B Function

4. What do we implement to restrict access to specific data rows of a calculation view?

   Choose the correct answer.

   - [X] A SQL Permission
   - [X] B Authorization Object
   - [X] C Analytic Privilege

5. Which SAP application is compatible with SAP HANA Live?

   Choose the correct answer.

   - [X] A SAP S/4HANA
   - [X] B SAP Business Suite
   - [X] C SAP Ariba

6. Which of these is used to build the virtual data model layer for SAP S/4HANA?

   Choose the correct answer.

   - [X] A ABAP CDS
   - [X] B SAP HANA Live
   - [X] C Universe
   - [X] D SAP HANA CDS

7. In SAP HANA Text Analysis, why would you use the *Voice of Customer* dictionary?

   Choose the correct answer.

   - [X] A To extract the sentiment from customer feedback on social media
   - [X] B To extract common entities such as company, country, currencies, and so on, found in documents
   - [X] C To identify close matches in words and expressions, to catch misspellings

8. Why would an organization implement SAP HANA Spatial?

   Choose the correct answer.

   - [X] A To enrich core ERP data with geographic insight
   - [X] B To identify unhappy customers who need a response.
   - [X] C To determine how closely related two customers are.

9. Which of these provide the algorithms that can be used with SAP HANA predictive analysis?

   Choose the correct answers.

   - [X] A EML
   - [X] B APL
   - [X] C PAL
   - [X] D SDA

10. SAP HANA Graph Processing powers real-time, high-performance business charts, and dashboards.

    Determine whether this statement is true or false.

    - [X] True
    - [X] False

Learning Assessment - Answers

160

1. What is the role of core modeling in SAP HANA?

   Choose the correct answers.

   [X] A To push data-intensive processing away from the application and to the database

   [X] B To develop re-useable data processing logic in the database

   [X] C To simplify application code

   Correct!

2. What are valid types of calculation view?

   Choose the correct answers.

   [X] A Dimension

   [X] B Cube

   [X] C Cube with star schema

   [X] D Dimension with star schema

   Correct! .

3. If I need to create multiple output data sets using SQL Script. Which HANA object do I use?

   Choose the correct answer.

   [X] A Procedure

   [X] B Function

   Correct! .

4. What do we implement to restrict access to specific data rows of a calculation view?

   Choose the correct answer.

   - [X] A SQL Permission
   - [X] B Authorization Object
   - [X] C Analytic Privilege

   Correct!

5. Which SAP application is compatible with SAP HANA Live?

   Choose the correct answer.

   - [X] A SAP S/4HANA
   - [X] B SAP Business Suite
   - [X] C SAP Ariba

   Correct!

6. Which of these is used to build the virtual data model layer for SAP S/4HANA?

   Choose the correct answer.

   - [X] A ABAP CDS
   - [X] B SAP HANA Live
   - [X] C Universe
   - [X] D SAP HANA CDS

   Correct!

7. In SAP HANA Text Analysis, why would you use the *Voice of Customer* dictionary?

   Choose the correct answer.

   - [X] A To extract the sentiment from customer feedback on social media
   - [X] B To extract common entities such as company, country, currencies, and so on, found in documents
   - [X] C To identify close matches in words and expressions, to catch misspellings

   Correct!

8. Why would an organization implement SAP HANA Spatial?

   Choose the correct answer.

   - [X] A To enrich core ERP data with geographic insight
   - [X] B To identify unhappy customers who need a response.
   - [X] C To determine how closely related two customers are.

   Correct!

9. Which of these provide the algorithms that can be used with SAP HANA predictive analysis?

   Choose the correct answers.

   - [X] A EML
   - [X] B APL
   - [X] C PAL
   - [X] D SDA

   Correct!

10. SAP HANA Graph Processing powers real-time, high-performance business charts, and dashboards.

    Determine whether this statement is true or false.

    - [X] True
    - [X] False

    Correct!

UNIT 4 Data Management with SAP HANA

UNIT OBJECTIVES

- Describe SAP HANA data management
- Describe SAP HANA data acquisition solutions
- SAP HANA Data Management Suite

Unit 4
Lesson 1

Describing Data Management

165

LESSON OVERVIEW
In this lesson we will explain what is meant by data provisioning with SAP HANA.

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Describe SAP HANA data management

Data Management in SAP HANA

Data management in SAP HANA covers all topics related to data acquisition and data storage. One of the key strengths of SAP HANA is its powerful data connection capabilities. With SAP HANA you can build innovative applications that can access any data of any type, anywhere. in real time or batch.

SAP HANA is able to extract and store the captured data but it can also create connections to remote data sources so that remote data source can be exposed to SAP HANA for live data access.

You need to decide which data acquisition tools are required. There are many to choose from and each has it strengths. For example, some tools have their strengths in complex ETL, whereas other tools can handle optimal fast real-time replication. Some tools can even handle a combination of both. For data that is loaded to SAP HANA, you also need to decide where the data resides. There are multiple options for specific data locality in SAP HANA that are suited to the different data temperatures.

You should pay special attention to the architecture of your data management solution in SAP HANA and design this well for optimum performance at the lowest cost, especially for high volume data scenarios that will grow over time.

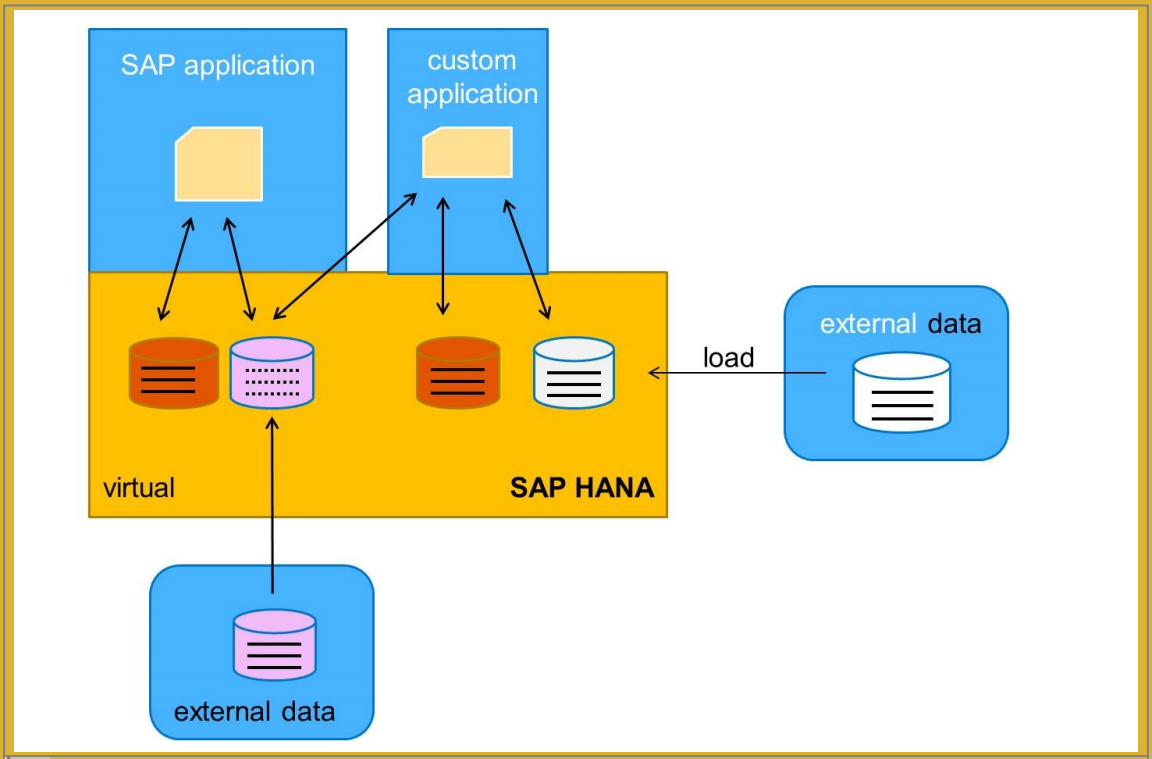


Figure 72: Data Acquisition

Before we dive into the data acquisition tools, let's look at a scenario where data acquisition tools may not even be needed.

Applications that are powered by SAP HANA generate SQL statements that create, update, and delete records in the HANA database. In this case, no additional data acquisition tools are needed. This is true for SAP applications such as S/4HANA and BW and also for custom applications that run on HANA.

However, it is important to understand what happens when SAP HANA is deployed as a standalone database, for example, as a data mart with analytics running on top. Even if SAP HANA is powering applications such as S/4HANA or BW, you may want these applications to have access to additional data sources, for example adding textual data from Twitter to a sales opportunity created in SAP ERP. This is why you implement the data acquisition tools of SAP HANA.

You should emphasize that SAP HANA is often used in a standalone data mart scenario where it is not powering an SAP application but is used as a store of data to support SAP BI use cases. In that case, data acquisition is essential because it is the only means to provide data to SAP HANA.

Data acquisition is not just another term for data loading. Data loading implies that data is physically moved and stored in SAP HANA. While this is possible, data acquisition also includes other approaches, such as data streaming and data virtualization. With these approaches, data is exposed to SAP HANA for processing, but data is not physically stored in SAP HANA. Another term that is being more frequently used in place of data acquisition is 'data ingestion'.

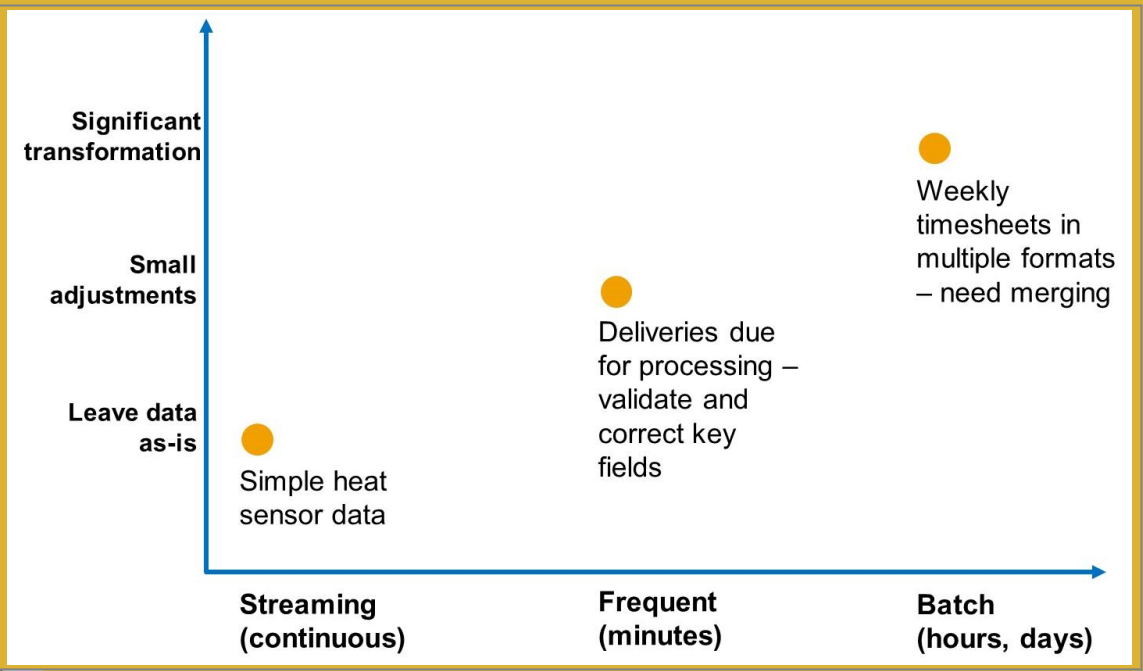


Figure 73: Aspects of Data Acquisition

Here is a great opportunity to ask students for other ideas for data acquisition scenarios and try to position them into this grid. Remember, this grid shows only three examples in three intersections. Make sure you emphasize that examples can appear in any intersection of this grid, not just along the illustrated corner to corner line.

When you consider data acquisition, think about the following key aspects:

- Frequency of data acquisition
- Transformation requirements of data

Frequency is represented on the horizontal axis of the figure, *Types of Data Acquisition*. Data can arrive at different time frequencies, ranging from real-time and hourly, to weekly, or yearly. It could also be driven by events, such as when a vending machine runs out of stock and transmits a request for a refill.

Transformation is represented on the vertical axis of the figure, *Types of Data Acquisition*. During provisioning, data can be transformed. This transformation could be done to align billing codes from different systems, convert currencies, look up missing zip codes, or calculate rebate values. Sometimes data from different sources must be loaded at the same time, for technical or business integration logic to be applied. An example of this is when data needs to be harmonized into a single stream from different sales order entry systems.

Data Provisioning Tools

SAP HANA allows any combination of provisioning frequency with any degree of transformation. This enables you to meet the needs of all applications that require data at any speed and of any type.

Today's modern applications are powered by a rich variety of data types (transactional, spatial, and text). These applications consume data at different rates from continuous real-time sensor data, to periodic batch loads of bulk data.

The course, HA350, *HANA Data Provisioning* covers various data provisioning tools in more detail. HA355, *SDI and SDQ* provides a deep dive into SDI and SDQ.

Data Storage

As SAP HANA evolves, there are many new options for data storage across tiers. You can decide, based on the importance of the data and performance expectations, exactly where data should reside, in or even outside HANA. There are many choices available for your data storage. You can choose between memory, disk, or external data archives. You can decide to implement Extended Storage or perhaps to implement multistore tables that cross tiers. You can classify your data by temperatures and assign the various temperatures to physical storage options in HANA.

Even though there is now vast amounts of memory available, it does not make sense to store old, rarely accessed data in memory. Data has a lifecycle and older data should be moved away from memory (hot), down to disk (warm), and eventually to archive (cold). The main reason for this is to ensure that you have the best possible performance for the data that is most used. You do not want to clog memory with huge amounts of old data that never gets used.

Data Temperatures

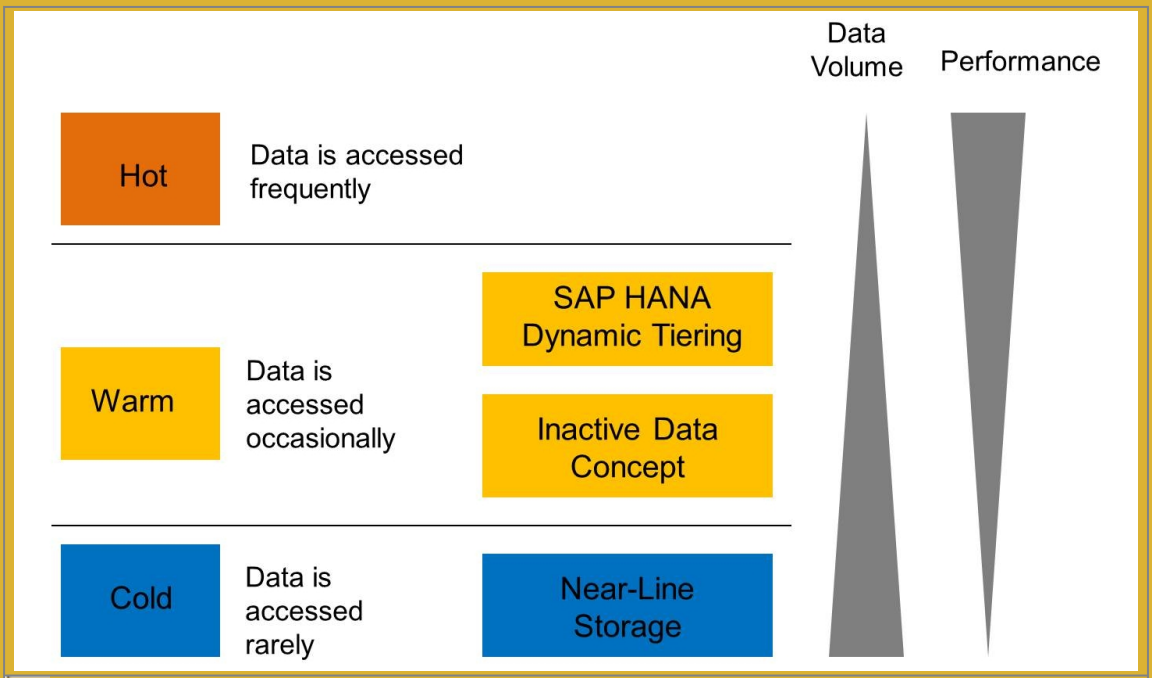


Figure 74: Data temperatures

You should size HANA memory according to hot data usage. This is an important calculation that can ensure that you get the right performance at the right price. You implement cheaper tiers such as disk and archives for the older data.

You also need to think about SLAs, that is, the performance requirement for different types of data. There is so much data acquired today that you need the think carefully about where to store this data.

Hot data is accessed frequently and/or needs very high performance. The storage for this is HANA memory.

Warm data is accessed less frequently and/or does not need high performance. The storage for this is disk, either via Extended Storage or inactive data (data that has been displaced from memory and pushed to disk).

Cold data is accessed rarely and/or does not need high performance. The storage for this is outside HANA, perhaps Near Line Store (NLS).

It is possible to classify and move data across the storage tiers automatically. Usually, it is the job of the application to control this based on usage patterns. For example in BW, you classify PSA as disk (warm) store, as this data is only needed periodically and should not occupy memory (hot).

It is also possible to control the movement of data across tiers under the control of HANA. For example, before running a job you can execute a command to load data to memory from disk. Then, once the job has ended, unload the data back to disk to clear out memory.

LESSON SUMMARY

You should now be able to:

- Describe SAP HANA data management

Unit 4
Lesson 2

Describing Data Acquisition Tools

170

LESSON OVERVIEW
This lesson describes the big picture of data provisioning in an SAP HANA landscape. The lesson will help the learner to understand the various scenarios for data provisioning.

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Describe SAP HANA data acquisition solutions

Data Virtualization

Customers have to deal with complex system landscapes across different locations, storing huge amounts of data in different formats and on different platforms. Customers require a cost-efficient, easy-to-deploy solution, to get real-time data visibility across their fragmented data sources. Examples of these include operational reporting, monitoring, predictive analysis, and transactional applications. With SAP HANA, we have built-in tools that can be used to create connections to remote data sources. This means that the data can be accessed in real time as required, as if it was actually stored in an SAP HANA database.

Smart Data Access (SDA) is the name of the built-in tool set that provides an extensive catalog of adaptors to connect to remote sources. SDA can figure out on-the-fly whether a query should be pushed down to the remote data source for processing, or whether the raw data should be fetched, and the query then runs in SAP HANA. SDA always uses the approach that offers the best performance.

Motivation for Smart Data Access

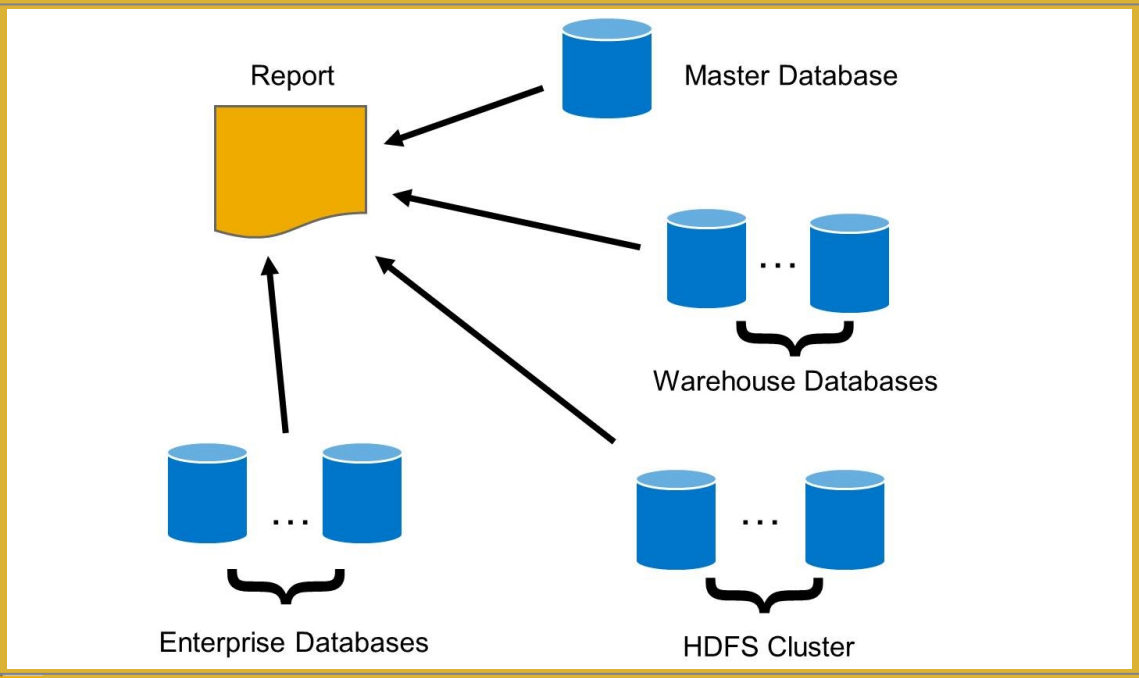


Figure 75: Motivation for Smart Data Access

To the application developer, the data appears to come from one source; that is, SAP HANA. Once the one-time connection to the remote source is established by IT, the application developers do not need to concern themselves with the technicalities of where the data is coming from.

SDA supports a modern data-federation strategy, where movement of data is minimized, and global access is enabled to data stored in local repositories.

SDA can be utilized in the following situations:

- To build new applications on SAP HANA, but access data from multiple sources without moving data into SAP HANA.
- To expose Big Data stores such as Hadoop to SAP HANA-based applications.
- To provide real-time access to archived data.
- To combine data from multiple SAP HANA-based data marts.

SAP HANA Smart Data Access: The Glue of the Platform

You can now create a fast and flexible data warehouse without expensive ETL or massive storage, security, and privacy risks. You can build Big Data applications with fast and secure query access to data, while minimizing unnecessary data transfers and data redundancy. You can bring social media data and critical enterprise information together, giving comprehensive insight into customer behavior and sentiment.

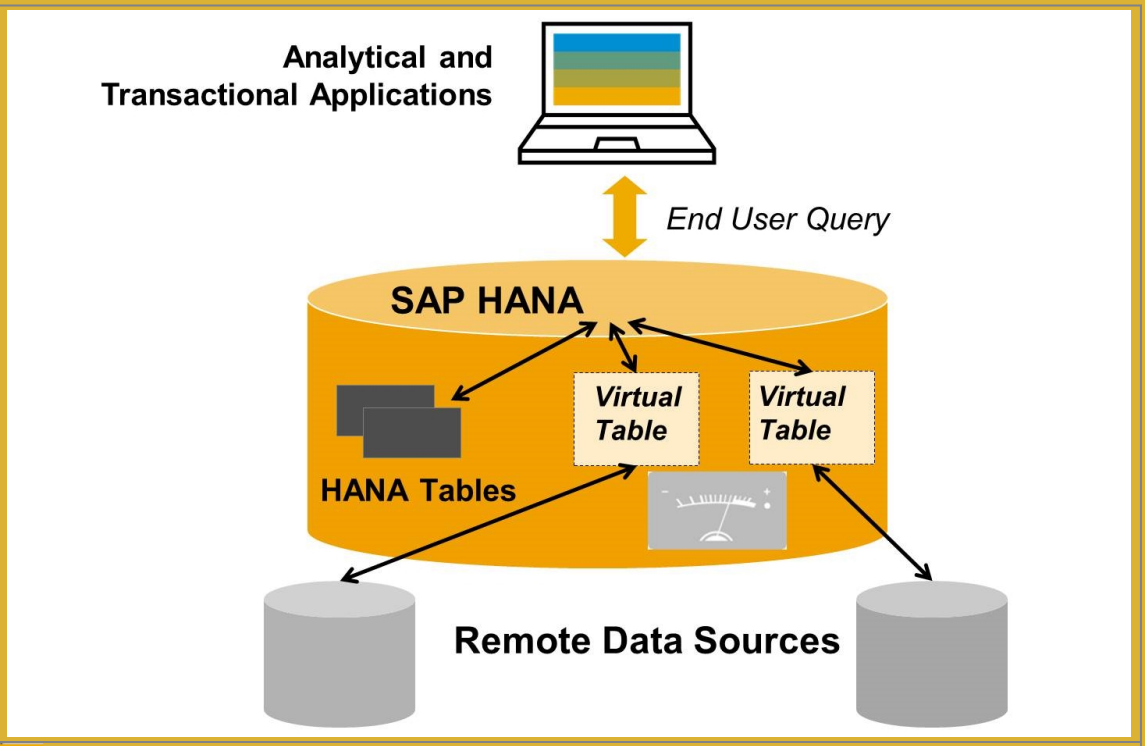


Figure 76: SAP HANA Smart Data Access: The Glue of the Platform

Benefits of SDA

SAP HANA SDA enables remote data to be accessed via SQL queries as if they were local tables in SAP HANA, without copying the data into SAP HANA. Specifically, in SAP HANA, you create virtual tables that point to remote tables or views in different data sources. Customers can then write SQL queries in SAP HANA, which can operate on these virtual tables. The virtual tables sit alongside the regular tables in the same SAP HANA database schemas. Although we are focusing here on the use of SDA to read data from remote sources, it should also be noted that SDA is able to write data to the remote source also.

Note:
A virtual table can easily be spotted in the database catalog as it has a small connector symbol added to the table icon.

The SAP HANA SQL query processor optimizes the queries that are based on virtual tables. It does this by determining if any query operations could be faster if they were pushed down to the remote source rather than processing in SAP HANA. To support this invisible decision making, SAP HANA collects statistics on the remote data sources.

The following are some of the benefits of SDA:

- Enables access to remote data just like local table
- Smart query processing pushes as much processing as possible to target data source
- Smart query processing includes query decomposition with predicated push-down, and functional compensation
- Automatic data-type translation enables remote data types to be mapped to HANA data types on the fly

- Supports data location agnostic development
- No special syntax to access heterogeneous data sources
- Provides SAP HANA to SAP HANA queries
- Supports Insert, Update, and Delete in many cases
- Calculation view support for Virtual Tables
- Delivers Generic Adapter framework to extend additional Remote Sources

If metadata changes in the remote database, it can be easily resynced with the HANA virtual table. This resyncing is done with no disruption to dependent objects (this feature was introduced with SAP HANA 2.0).

Virtual tables also cache their results so that identical queries do not have to fetch the same data again. The cached data, if unexpired, can be used again, which vastly improves performance (this feature was introduced with SAP HANA 2.0).

Note:
For the up-to-date list of adaptors, check SAP Note 1868209.

For detailed coverage of SDA, refer to the training course HA350 where we set up virtual tables.

Data Replication

Data replication typically means ensuring that data created in one system is duplicated to one or more target systems. It is usually done in real time and is managed record by record. However, replication does not always happen in real time. Replication can also take place periodically, for example, every five minutes. Periodic replication is used when it is not essential that data is always synchronized in real time.

Typically, with replication, no transformation takes place on the data as it travels to the target system so that we have an identical copy of the data in all systems. Replication involves the physical moving and loading of data, and not simply exposing the data sources as virtual tables.

The following examples illustrate data replication:

- Sales orders entered into various SAP ERP systems are immediately replicated to a central data mart. This means real-time dashboards can be developed to show the live sales pool.
- A popular vending machine sends its stock information every ten minutes to the central inventory system at HQ. This can trigger replenishments when stocks are low.
- Orders are collected centrally, and simultaneously a copy of the order is routed immediately to the relevant warehouse for processing.

There are many different technical implementation approaches that support replication, ranging from the use of database logs to the use of database triggers. It is essential that the source or target system has some way of knowing that data has changed, so that replication can be kicked off.

The SAP HANA real-time replication solutions provide technologies for replicating data, in real time or batch, from any source system to the SAP HANA database. These include trigger-based data replication using **SAP LT Replication Server (SLT)**, log-based data replication

technology using **SAP Replication Server**, and session-based synchronization using **SAP HANA Remote Data Sync.**

SAP LT Replication Server (SLT)

SAP LT Replication Server (SLT) is a popular tool used by customers to ensure that data generated in an SAP Business Suite application is replicated in real time to SAP HANA. One of the main use cases for SLT is to provision data to SAP HANA for BI cases. For example, live dashboards can be kept up-to-date with real time transaction data.

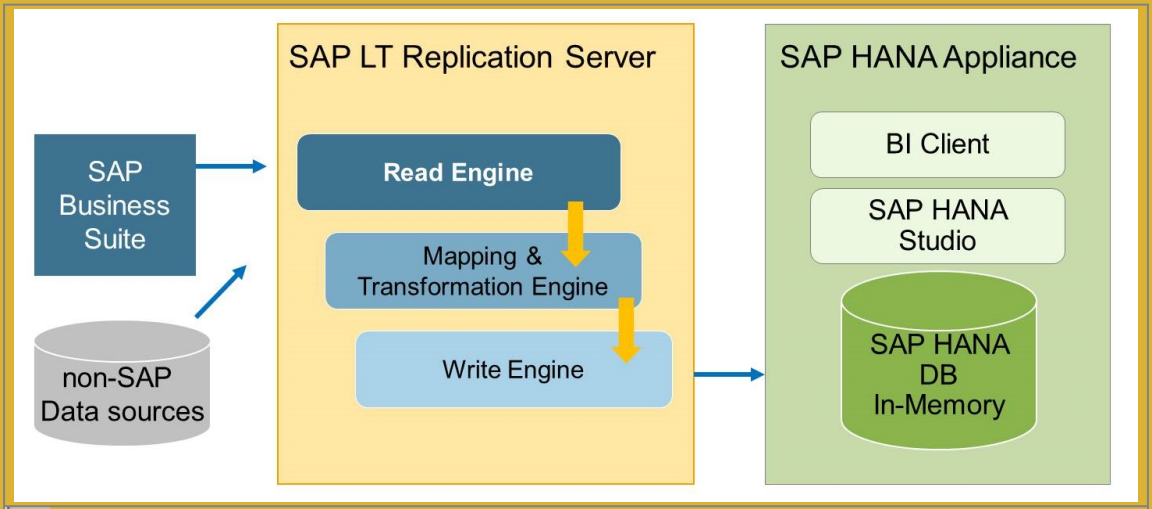


Figure 77: SAP LT Replication Server

SLT is an SAP NetWeaver ABAP-based application and it uses well known SAP technologies such as RFC and DB Connect to establish source and target connections. SLT is also used as the data transfer tool for many SAP products, such as SAP BW and SAP Accelerators. SLT plays a key role when SAP HANA Live is deployed as a side car, by managing all data replication.

SLT has been used for many years as a data transfer tool in landscape transformation scenarios (company acquisitions where data needs to be consolidated, or split). SLT predates SAP HANA. In the last few years, SLT has had many enhancements specifically aimed at its use with SAP HANA as a replication tool. Many of these enhancements help to improve the throughput of data as well as the monitoring of the data movement.

SLT is a trigger-based data provisioning tool. This means that SLT sets database triggers on any table that you want to replicate from.

When the database table is changed in any way (insert, update, or delete), the trigger is automatically fired. SLT hears the trigger and fetches the data from the source system and transfers it to SAP HANA.

SLT can perform the following types of data movement:

- Load

  This is a one-time data copy from the source system to SAP HANA. This is not replication, but a bulk copy. This tool is also used for data migration, which is typically a one-time event, so this feature is very important.

- Replication

SLT performs an initial full load of all data, and then immediately sets up a real-time replication of data from the source to the target system. This replication is trigger-based, meaning that a DB trigger is defined in the source database on each table marked for replication. Each time a data modification is done to a source table, the trigger captures this event and SLT transports this change to the target database. This is typically the most popular use of SLT, to enable feeding of real-time data to SAP HANA-based applications.

When SAP HANA tables are the target, SLT replicates data from the source system at the table level. Some data provisioning tools are able to replicate from the application level using business views (for example, BW data sources). This means that you need to know the names of the source tables you wish to replicate from.

When we think about replication we usually assume that data moves unchanged from the source to the target. However, in some cases, you may need to apply some transformation to the data.

Although SLT is not a heavyweight data transformation tool, it is possible to modify the data during the transfer process. The types of possible modifications are as follows:

- Add or remove table columns

  For example, with a wide transactional record, perhaps only a few columns are needed for reporting, or perhaps a new column to store a new calculation is needed.

- Change data type for a column

  For example, the source column is an integer type and you wish to convert this to a character string so it is able to hold more flexible values.

- Filter

  For example, you only need to replicate the orders flagged as ‘URGENT’. The filters can be set on multiple columns.

- Modify data

  For example, you wish to convert column values to align to and with other systems (for example, you load records with country code ‘GER’ but they need to be converted to the corporate standard ‘DE’).

- Split up a table

  This is not strictly a modification, but SLT enables the distribution of data from a single source table to multiple SAP HANA target tables (and the other way around).

ABAP is used to develop the transformation logic. So, this is a crucial skill to have on any SLT project where transformations are made.

Any transformation applied to data as it is being replicated has an effect on the time it takes for the data to arrive at the target. For this reason, only light transformations are implemented.

Writing data transfer rules for complex integration and cleaning can get complicated. There are better SAP data provisioning tools to use in those situations.

The administration of replication using SLT is fully integrated into the SAP HANA Studio. Here, you can choose from a number of options to stop and start data movement jobs.

SLT is a key tool used with SAP HANA Live side-by-side scenario, as well as Central Finance with SAP S/4HANA.

Note:
You can learn more about SLT in the SAP course, *SLT100*.

SAP Replication Server

SAP Replication Server is a sophisticated transactional data movement product that moves and synchronizes data across the enterprise. This is done without geographical distance limitation to meet demanding requirements in the enterprise such as guaranteed data delivery, real-time business intelligence, and zero operational downtime. SAP Replication Server facilitates this by non-intrusively handling data at the source and target database level, while ensuring high performance and transactional integrity.

SAP Replication Server is often used by organizations that need to move a lot of data in different directions in real time and ensure 100% synchronicity. You will find this solution used in many financial institutions where systems must be completely in step, in real time, with robust recovery options in case of failure.

SAP Replication Server was a well known tool acquired by SAP from Sybase. This is why it is still often referred to as Sybase Replication Server, or SAP Sybase Replication Server — both of these are now invalid names. Also, SAP Replication Server is a powerhouse for data distribution management, and as a result is complex and requires significant skills to set up and administer. This is why we tend to not focus on this for SAP HANA, where much simpler options exist. SAP Replication Server is more than capable of providing replication services to SAP HANA, but it is regarded as overkill for most SAP HANA use cases.

Some of the key benefits of SAP Replication Server are as follows:

- Log-based replication process
  - Non-intrusive
  - Very high performance
- Improve recovery, resumption times and minimize downtime
  - Bidirectional replication
  - Standby DB is always available and can be used for read-only report server
- Fresh data to enable a timely decision
  - Run resource-intensive reports on reporting servers without impacting OLTP systems
  - Reduce information latency for reporting and optimize batch reporting
- Real-time data sharing and synchronization
  - Facilities decentralized business operations
  - Enables remote applications to access data locally for improved performance

One key feature of SAP Replication Server is that it relies on a log-based replication technique. The Changed Data Capture (CDC) is not done against the data volumes of the source database tables, but instead by reading directly from the database log.

Note:
A database log is a history of all actions executed by the database management system. A log is often used in the recovery of databases after a crash. When replayed, all updates to the database can be re-created.

This log-based approach reduces the workload that the replication process usually brings to the source database, thus enhancing the availability of this system.

SAP Replication Server has been enhanced recently to support replication scenarios that include SAP HANA as a source or target.

Among the latest enhancements is the ability to replicate from Business Suite applications, as SAP Replication Server is now able to handle the SAP proprietary ABAP cluster tables.

While SAP Replication Server can handle large volumes of non-disruptive replications, there are other SAP HANA replication options that can achieve satisfactory results with a much simpler landscape.

Note:
For more details on SAP Replication Server, refer to the SAP course *EDB374*.

SAP HANA Remote Data Sync

SAP HANA remote data sync is a synchronization technology designed to synchronize remote databases with a central database. SAP HANA is the central database and the remote sources can be either of the following:

- SAP SQL Anywhere

  This is a powerful, highly scalable relational database that can be deployed in large or small remote applications.

- UltraLite

  This is a component within SAP SQL Anywhere used to support small footprint applications found in handheld devices such as smartphones or tablets.

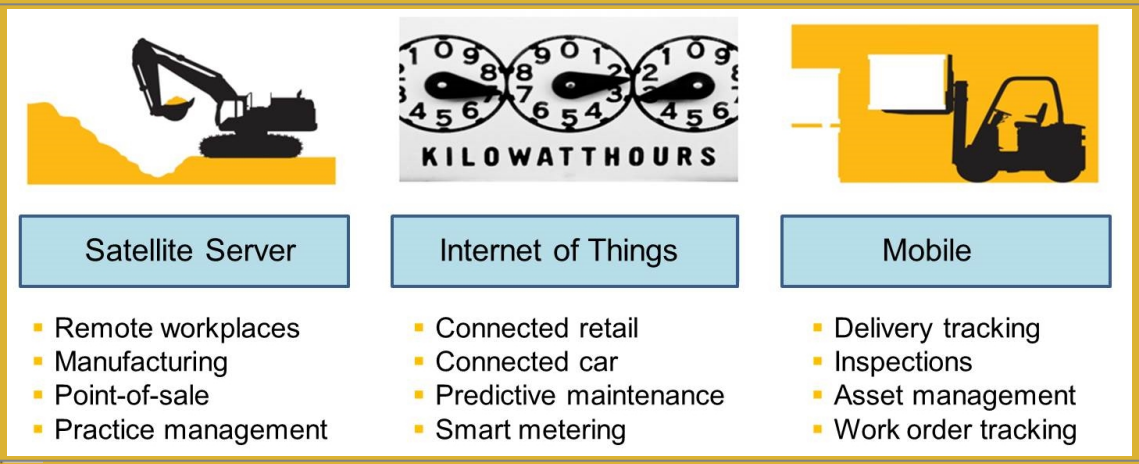


Figure 78: SAP HANA Remote Data Sync

SAP HANA remote data sync is useful when applications cannot remain continually connected to a central database due to connection problems. An example of this could include a field engineer in a remote location with a poor signal, or perhaps if the application should not be continually connected due to connection costs.

When implementing SAP HANA remote data sync, you develop occasionally connected, smart client applications. These applications sync with the central database, either periodically at set times or triggered by an event.

In all remote data sync applications, the remote data sync server is the key to the synchronization process. Synchronization typically begins when a remote data sync remote site opens a connection to a remote data sync server. During synchronization, the remote data sync client at the remote site can upload database changes that were made to the remote database since the previous synchronization. On receiving this data, the remote data sync server updates the consolidated database, and then downloads changes from the consolidated database to the remote database.

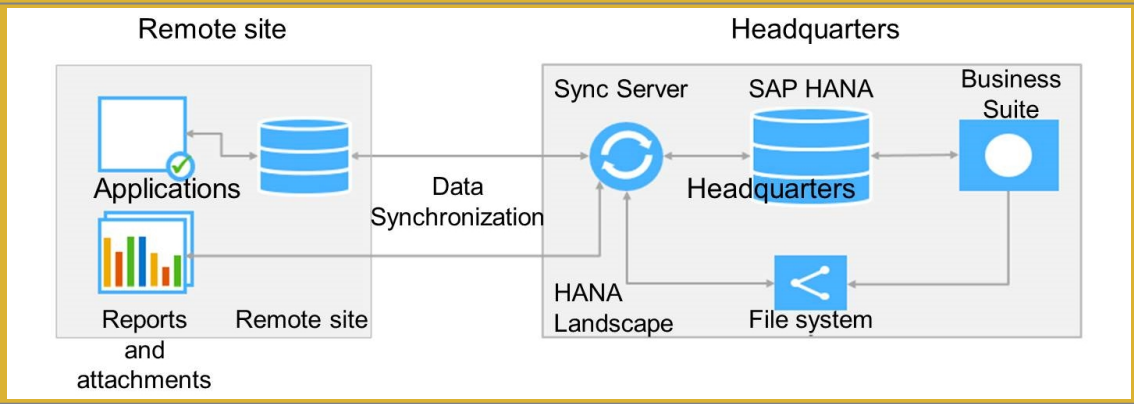


Figure 79: Remote Data Sync Architecture

One of the key advantages of SAP HANA remote data sync is that it maintains data integrity at all times. It remembers the exact sequence of updates from all remote clients. Imagine for example if 1,000 field engineers were withdrawing the same spare part, while at the same time other remote works were replenishing the same spare part. It could be easy for the stock balances to get messed up in fast, bidirectional data traffic. Fortunately, SAP HANA remote data sync handles the updates in sequence, with 100% accuracy.

Extract, Transform, Load (ETL)

Extract, Transform, Load (ETL) is the process of extracting data from source systems and applying transformations on the data, before loading to a target. This process is popular with data warehouses, such as SAP BW where there are many data sources that need merging. SAP HANA has two key tools that can be used in the data provisioning scenario around ETL. These are SDI/SDQ and SAP Data Services.

Smart Data Integration (SDI)/Smart Data Quality (SDQ)

There are many options to choose from when considering data provisioning to SAP HANA. Most of these options require the installation and set up of additional software and hardware components, which sit between the data source and SAP HANA. These components cover a broad range of capabilities, including extracting data, combining sources, and cleansing, loading, or exposing the data to SAP HANA.

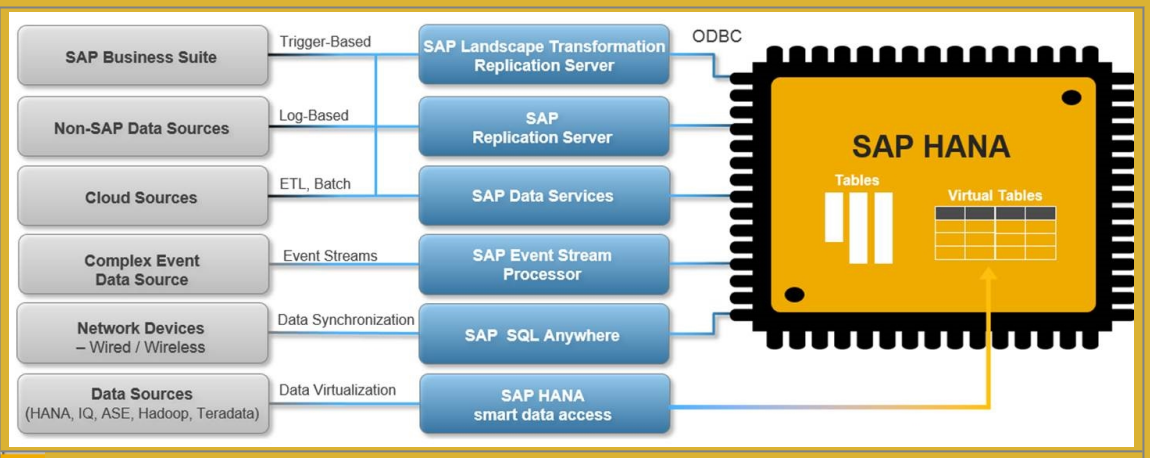


Figure 80: Traditional Approach to EIM

The inclusion of an additional data provisioning tier adds to the complexity of the overall solution.

A Simplified Approach to Data Integration

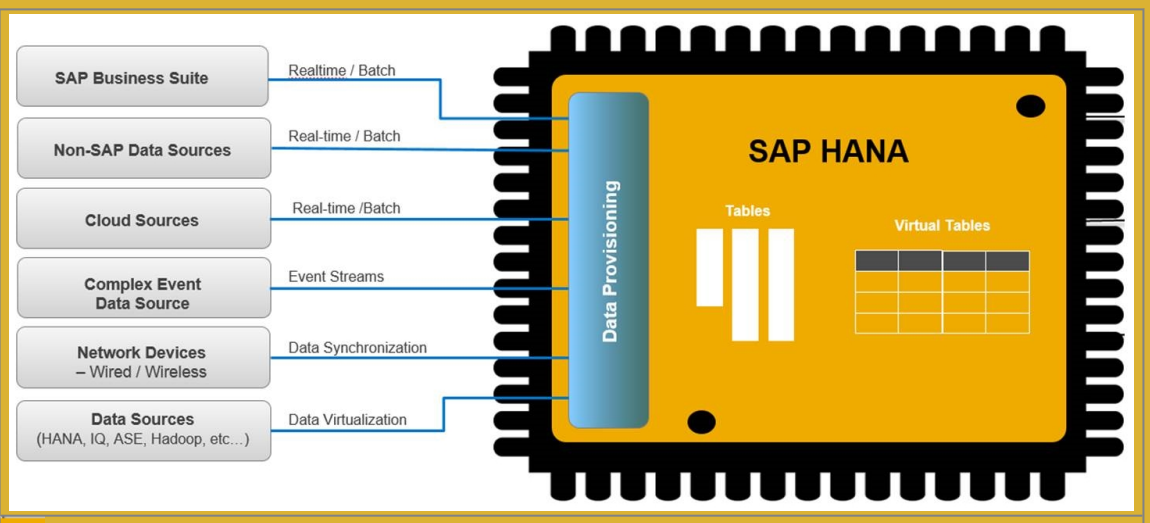


Figure 81: Simplified Approach to EIM

SAP HANA has its own built-in ETL capabilities. We call this Enterprise Information Management (EIM). The two components of EIM are Smart Data Integration (SDI) and Smart Data Quality (SDQ). This means that no additional tools and associated hardware are required, as everything you need is provided with SAP HANA.

With EIM, we have removed the external data provisioning tier. Running data provisioning tasks inside SAP HANA also means that we take advantage of the high performance, in-memory processing for data acquisition tasks.

Components of EIM

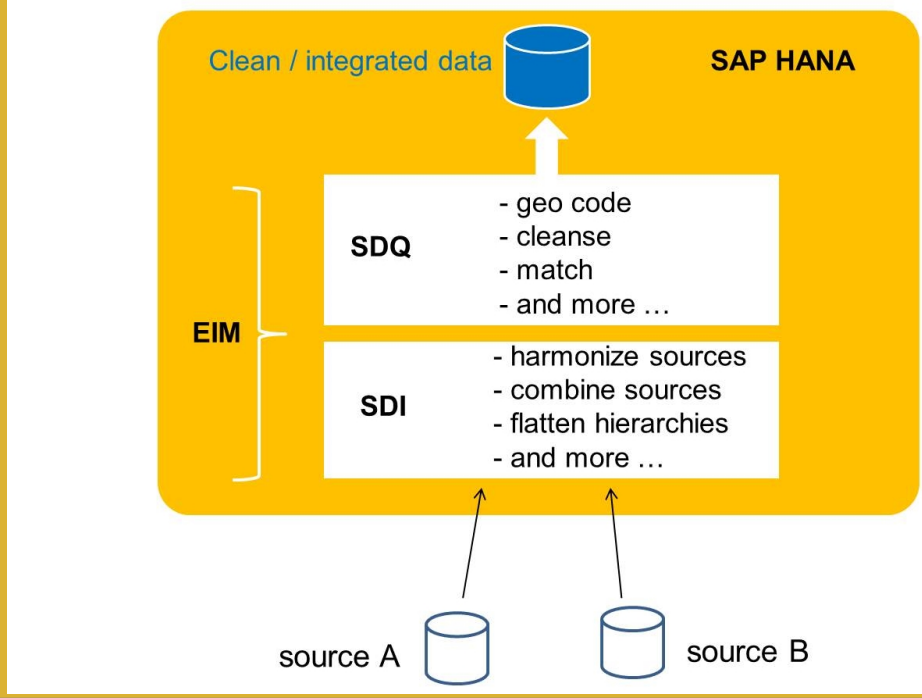


Figure 82: Components of EIM

The following are the components of EIM:

- Smart Data Integration (SDI)

  Functions for acquiring and integrating data from multiple sources

- Smart Data Quality (SDQ)

  Functions for improving data quality

Although we see two components, SDI and SDQ, do not think of these as two separate products. When building any data provisioning job, the developer is able to include any of the capabilities from either component. SDI is the key component that takes care of data acquisition and integration, whereas SDQ can add additional steps to the job to enhance and clean the data. SDQ relies on the basic features of SDI to get the data moving.

- Transformations enriching data
- Cleanse
  Parse, standardize and enrich person, title, phone, firm, email and address information within a specified input source.
- Geocode
  Enrich address data with associated latitude and longitude information

- **Simplify cleanse transforms:**
  Single transform deals with
  - Person names and titles, phone, email,
  - Firm,
  - Address information

  In Data Services, it is in 2 transforms
- **Consolidate available configuration options:**
  Improved productivity at functional parity

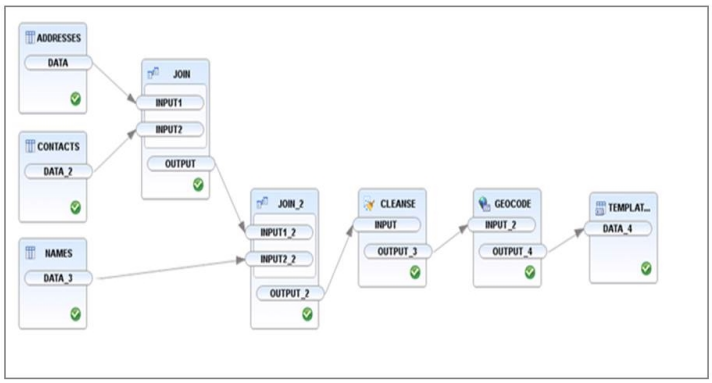


Figure 83: Building an EIM data flow

EIM jobs are created in SAP HANA using flowgraphs. Flowgraphs are graphical representations of a data provisioning job. They contain a sequence of nodes and flow lines that represent the steps in the flow. Developers create jobs by dragging and dropping the nodes to a canvas to create the flowgraph.

Students who are familiar with SAP Data Services recognize many of the features of EIM. With each new release of SAP HANA, the EIM capabilities align more with Data Services. However, for now Data Services is still the power house solution for complex ETL and data profiling services. The key point here is that we want to offer a way to simplify the landscape with more and more native built-in capabilities of SAP HANA taking over where additional software components were needed. This is also important for the cloud solutions where we need fully-integrated capabilities for ETL.

SAP Data Services

SAP Data Services provides the capability to extract, transform, and load (ETL) data from any source and to any target. This includes SAP HANA as both a source or target. SAP Data Services has been around for many years and is deeply embedded in the distributed landscapes of many customers. So for those customers it is good news that SAP HANA can also be integrated as a data source or target in a mixed landscape.

SAP Data Services is SAP's most powerful multi-purpose ETL platform. It does not rely on SAP HANA and operates on its own server. It provides very sophisticated data integration and harmonization features, as well as country-specific data cleansing tools.

SAP Data Services can pass and exchange metadata with SAP HANA. For example, the output structures of the Data Services jobs can automatically create SAP HANA tables. Or, SAP HANA tables and views can be automatically exposed to Data Services jobs for input.

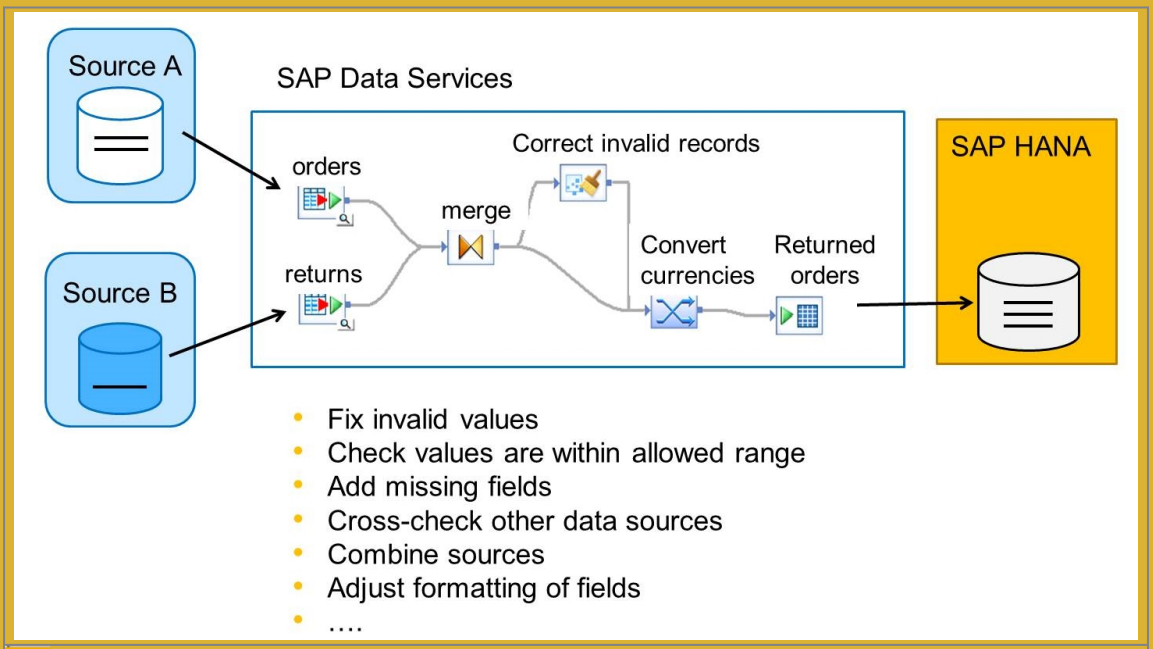


Figure 84: SAP Data Services

Core Capabilities of SAP Data Services

SAP Data Services usually processes all data in its own engines and sends the output to the target systems. When SAP Data Services is used with SAP HANA, a significant amount of the data processing is pushed to SAP HANA, to ensure ETL jobs run as fast as possible. SAP have enhanced Data Services recently to provide tighter integration with SAP HANA, so data flow jobs are optimized.

Although the core capabilities of SAP Data Services are also available in SDI/SDQ, SAP Data Services continues to provide many more data integration and quality transforms that are not yet available in SAP HANA SDI/SDQ. SAP Data Services excels at managing complex delta loading to data warehouses with auto-recovery mechanisms built-in to restart jobs if they fail. SAP Data Services can also quarantine data that does not pass quality checks for more intensive processing.

SAP Data Services is a family of tools and a key component is SAP Information Steward. This tool provides extensive data quality and data profiling dashboards so business users can monitor and measure data quality. The tool also exposes the data cleansing rules to business users who can create and adjust cleaning rules without the need for IT involvement. Enterprise-wide data lineage is also a capability of Information Steward so that the origins of data can be traced from reports.

Over time, we may find some or all of these capabilities appearing in SAP HANA. However, for now SAP Data Services remains a good choice for many customers who need a fully loaded ETL solution with sophisticated features for a complex data landscape.

Direct Extractor Connection (DXC) is also part of the ETL family for SAP HANA data provisioning. However, due to its lack of popularity we have decided to leave this out of the HA100 course.

Unit 4
Exercise 8

Extract, Transform, and Load Data Using SDI

183

Exercise Objective

In this exercise, you learn how to build a simple flowgraph within the context of SDI / SDQ in order to extract, enrich and load data to an SAP HANA tabe.

Business Example

Each week, HR department updates a simple flat file with recently joined employees. The file is located in a Windows folder. You would like to load the records from the flat file to an SAP HANA table. One of the business reports requires the gender of the employee to be identified but gender is not provided by the record. You would like to enrich the employee record by having SAP HANA determine the gender from the name of the person.

We have already created a lot of the setup for this exercise. Students locate, examine, and even add a new record to the source flat file but they don't get to see the corresponding virtual table and synonym (we tried to keep the exercise simple as this is supposed to be an overview class). It might be a good idea to show the class the remote source definition and the files it exposes and then explain how we created virtual tables in the TRAINING schema. (which could actually be accessed directly without a flowgraph). The synonyms were needed as we need to access the virtual tables that are not in our container, but in the external TRAINING schema. You might show these.

1. Examine the contents of the source flat file *consultants.csv* using Notepad, located in *Favorites → HA100*.
2. Add a new record to the end of the file making up a new employee name, as follows:
   `0007,any first name,any second name,70`
   Don't forget the three separator commas, and also don't forget to add a new first and second name.
3. In the Web IDE, create a new flowgraph with the name **`load_employees_##`**.
4. Add a node to the flowgraph to define a data source based on your virtual table *Consultants_##*.
5. Add a node after the data source node to cleanse the source data.
6. Connect the *Data Source* node with the *Cleanse* node.
7. Configure the *Cleanse* node to generate an extra column that identifies the gender of each employee.
8. Add a node to define a data target.
9. Connect the *Cleanse* node with the *Data Target* node.
10. Configure the *Data Target* to use a template table with the name *Employees_Gender_##*.

11. Save and build the flowgraph.
12. Execute the flowgraph.
13. Check the results in the newly created target table *Employees_Gender_##*.

Note:
If you want to re-run the flowgraph, you need to add the setting *Truncate Table* in the *Settings* of the Data Target node, save, and rebuild the flowgraph. Otherwise, the load fails with a key violation error due to duplicate keys.

Extract, Transform, and Load Data Using SDI

185

Exercise Objective

In this exercise, you learn how to build a simple flowgraph within the context of SDI / SDQ in order to extract, enrich and load data to an SAP HANA tabe.

Business Example

Each week, HR department updates a simple flat file with recently joined employees. The file is located in a Windows folder. You would like to load the records from the flat file to an SAP HANA table. One of the business reports requires the gender of the employee to be identified but gender is not provided by the record. You would like to enrich the employee record by having SAP HANA determine the gender from the name of the person.

We have already created a lot of the setup for this exercise. Students locate, examine, and even add a new record to the source flat file but they don't get to see the corresponding virtual table and synonym (we tried to keep the exercise simple as this is supposed to be an overview class). It might be a good idea to show the class the remote source definition and the files it exposes and then explain how we created virtual tables in the TRAINING schema. (which could actually be accessed directly without a flowgraph). The synonyms were needed as we need to access the virtual tables that are not in our container, but in the external TRAINING schema. You might show these.

1. Examine the contents of the source flat file *consultants.csv* using Notepad, located in *Favorites* → *HA100*.
   a) Expand the folder *Favorites* → *HA100*. Right-click the file *consultants.csv*, and choose *Notepad* to display the contents. (Do not double-click.)
2. Add a new record to the end of the file making up a new employee name, as follows: **`0007,any first name,any second name,70`**
   Don't forget the three separator commas, and also don't forget to add a new first and second name.
   a) Enter **`0007,any first name,any second name,70`** at the end of the file. Don't forget to add a first and second name.
   b) Choose *File* → *Save*.
   c) Close the file.
3. In the Web IDE, create a new flowgraph with the name **`load_employees_##`**.
   a) Expand the folders *HDB* → *src*.
   b) Right-click the folder *exercises* and choose *New* → *Flowgraph*.
   c) At the prompt, enter the name **`load_employees_##`**, and choose *OK*.

4. Add a node to the flowgraph to define a data source based on your virtual table *Consultants_##*.
   a) Choose  *Add Node* icon and from the list that appears, choose *Data Source*. Click anywhere to the left of the canvas to drop the node.
   b) In the *Data Source* node, choose  *configure* symbol at the bottom of the node.
   c) Choose *HANA Object* so the *Data Source* selector appears.
   d) In the search field, enter **cons** so that the virtual table *Consultants_##* appears in the list below.
   e) Highlight the row that contains the name of your virtual table *Consultants_##*, and choose *Finish*.
   f) Choose *Apply* to save and close the node.
5. Add a node after the data source node to cleanse the source data.
   a) Choose  *Add Node* icon and from the provided list, choose *Cleanse* and then click anywhere in the center of the canvas to drop the node.
6. Connect the *Data Source* node with the *Cleanse* node.
   a) Drag a line from the output port on the *Data Source* node to the input port *IN* on the *Cleanse* node.
7. Configure the *Cleanse* node to generate an extra column that identifies the gender of each employee.
   a) In the *Cleanse* node, choose  *configure* icon at the bottom of the node.
   b) Choose the *Edit Defaults* button and from the menu choose *Edit Content Types* to observe the suggested content types that SAP HANA has assigned to each column of the source data.
   c) Assign the column *NAME* to the data content type *Unknown* as this column is the source file name, not a person's name.
   d) Observe how the input columns *FirstName* and *LastName* have automatically been assigned the correct content types.
   e) Choose *Save*.
   f) Choose the circular button *Cleansed Output* and choose the button *Customize Manually*.
   g) De-select the two columns that have been automatically checked. You do not use these.
   h) Navigate to the component *Person* → *Person Extended* and then select the option *Gender (MALE_STRONG)* and choose *Save*.
   i) Choose the circular button *Output Summary* and review the final output columns that now include the generated column for gender.
   j) Choose the *Apply* button to save the node and return to the canvas.
8. Add a node to define a data target.

a) Choose  *Add Node* icon and from the provided list, choose *Data Target* and then click anywhere to the right of the canvas to drop the node.

9. Connect the *Cleanse* node with the *Data Target* node.
   a) Drag a line from the output port on the *Cleanse* node to the input port on the *Data Target* node.
10. Configure the *Data Target* to use a template table with the name *Employees_Gender_##*.
    a) In the *Data Target* node, choose  *configure* icon at the bottom of the node.
    b) Choose the *Template Table* button and in the *Object Name* field enter **`Employees_Gender_##`**.
    c) Choose *Apply*.
11. Save and build the flowgraph.
    a) Choose *Save*.
    b) Choose the menu option *Build* → *Build Selected Files* and check the log to ensure the flowgraph built successfully.
12. Execute the flowgraph.
    a) At the top of the flowgraph canvas you see a short toolbar. Choose the  icon. (When you hover over it you see the word *Execute*).

> 
> Caution:
> On the screen you see two icons that use the same triangle symbol. The one in the top main toolbar is used to *Run* XSA applications and not to *Execute* a flowgraph.

    b) At the confirmation prompt, choose *OK*.
    c) If it's not already open, open the *Console* to check the log. (The load takes about 3 seconds).
13. Check the results in the newly created target table *Employees_Gender_##*.
    a) Switch to the *Database Explorer* view.
    b) Expand the entry *My HA100 Container* → *Tables*.
    c) In the lower pane, right-click the table *Employees_Gender_##*, and choose *Open Data* and you should see the employee records. Each has a gender assigned.

> 
> Note:
> If you want to re-run the flowgraph, you need to add the setting *Truncate Table* in the *Settings* of the Data Target node, save, and rebuild the flowgraph. Otherwise, the load fails with a key violation error due to duplicate keys.

Data Streaming

Data streaming is the transfer and processing of continuous data from a source to a target. Data streaming often involves high-speed, high-volume data transfers from multiple streams in parallel. Sources of streaming data can range from simple sensors to complex business systems.

In today's highly connected digital world, data streaming is an essential enabler of real-time information, to feed applications and dashboards. The opportunities for the development of innovate applications are enormous.

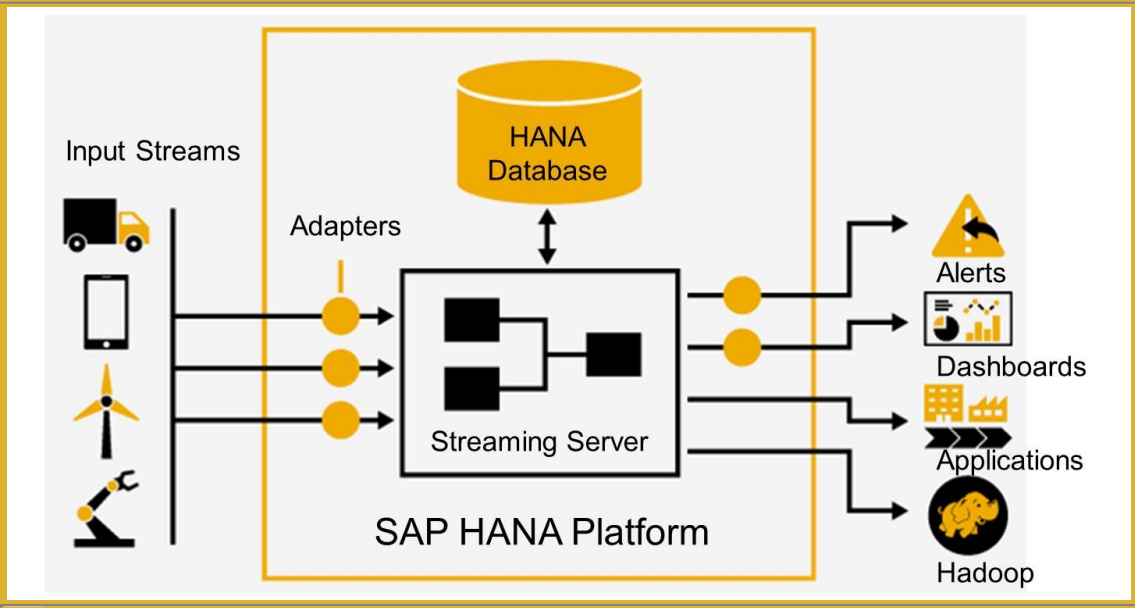


Figure 85: SAP HANA Streaming Analytics

Enterprises today are flooded with streams of notifications as things happen. Individual events may not be significant by themselves, which makes it difficult to discern when consequential events do occur. You could have thousands of sensors reporting statuses every few seconds, and most of that information is uninteresting. However, when something is starting to go wrong, you want to know as soon as possible, so that you can act before a small trend becomes a significant problem.

Data streams contribute significantly to the size of the world's digital data, known as Big Data.

SAP HANA Streaming Analytics

SAP HANA Streaming Analytics has its roots in an SAP product called Event Stream Processor (ESP), which was originally developed by Sybase. When SAP introduced data streaming to SAP HANA, it was called Smart Data Streaming (SDS). It has now been re-branded as **SAP HANA Streaming Analytics**.

The streaming analytics server runs as a separate server in the SAP HANA landscape, separate from, but interacting with, the SAP HANA database.

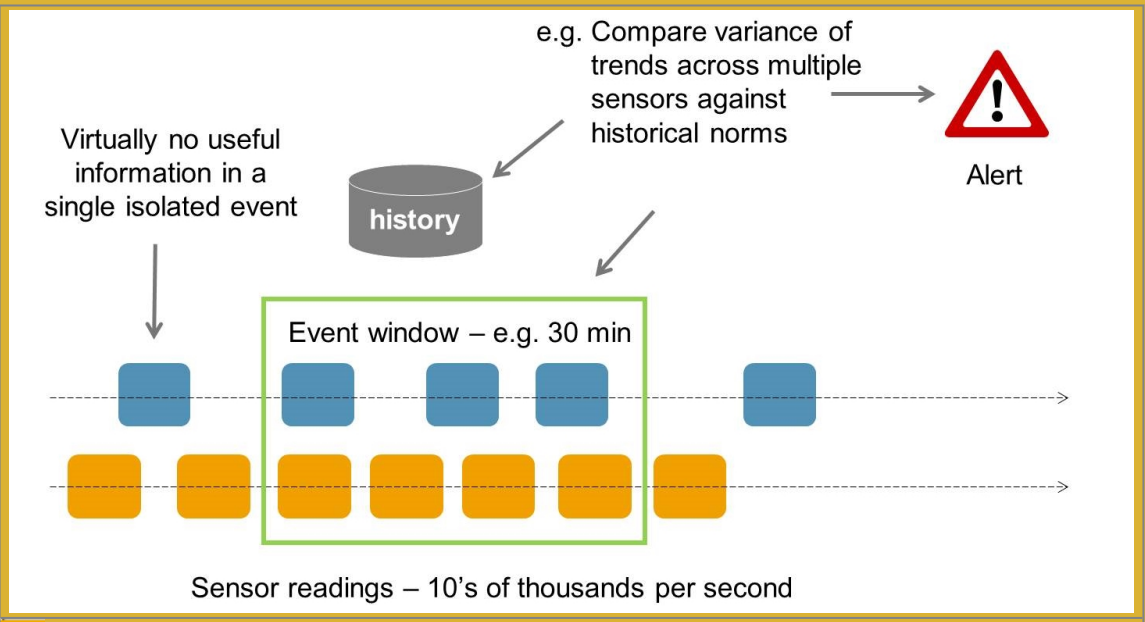


Figure 86: SAP HANA Streaming Analytics

With SAP HANA Streaming Analytics, you can capture data (millions of events per second) arriving continuously from devices and applications, and act on this new information as soon as it arrives. This enables you to react in real time using alerts, notifications, and immediate responses to continually-changing conditions. Often a single stream offers little useful information but when we combine multiple streams triggered by different events, we can begin to develop real insight.

Note:
SAP HANA Streaming Analytics is also known as SAP HANA Smart Data Streaming (SDS) and is covered in more detail in the SAP course, *HA350*.

Flat File Loading

One of the most popular ways data is presented for loading to SAP HANA is through a flat file. There are multiple methods to loading flat files into SAP HANA. One of the quickest and simplest ways to load data to SAP HANA is by using the flat file import wizard in the SAP HANA Studio. The wizard supports either csv files or Excel workbooks.

Using the SAP HANA Studio Import Wizard

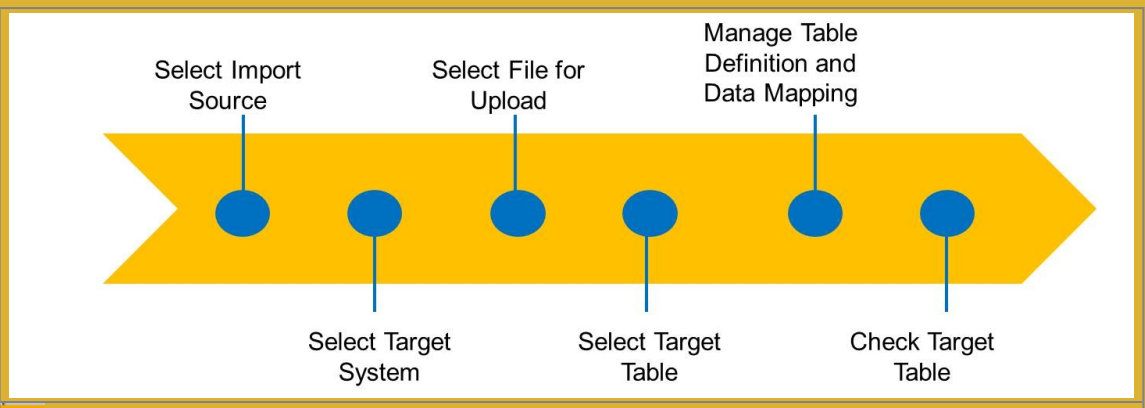


Figure 87: Process Flow: Uploading Data from Flat Files

The process to load a flat file to SAP HANA using SAP HANA Studio is simple and is guided by a wizard. The process is as follows:

1. From the SAP HANA Studio *Import* menu, choose the option to import data from a local file.
2. Choose the target SAP HANA database.
3. Choose the flat file to import.
4. Choose the existing SAP HANA target table to load to, or choose to automatically create a new table.
5. Confirm or adjust the suggested field mapping and column types.
6. Preview the output and if you are happy, execute the load.

Flat File Import Wizard — What You Can and Can't Do

To upload data from flat files, SAP HANA Studio flat file import wizard offers the following features:

- You can load to an existing SAP HANA table or have the wizard create the table for you.
- If the target table does not exist in SAP HANA database, the wizard creates the empty table automatically based on whatever metadata can be supplied or determined from the flat file.
- The wizard suggests the column name and data type for the new tables. It is possible to edit these suggestions.
- The new table always has a 1:1 mapping between the file and table columns. You can't split columns up or concatenate.
- You can add extra empty columns or remove columns from the source file that you don't need.
- You cannot define filters. All records are loaded.
- When loading new data to an existing table, it is appended to any existing data. Existing records are never overwritten — so make sure that the new records do not create key violations.
- When loading data to an existing table you cannot overwrite the target table column names or change the data types and lengths.
- The supported file types are .csv, .xls, and .xlsx.

You could mention that there is no equivalent tool in Web IDE, and that is why we are hanging on to this great feature of SAP HANA Studio. To load flat files in Web IDE you would manually import the flat file to your project folder and then define a loading instruction file (.hdbtabledata) that identifies the source location/filename and also the target table to load and the file parameters. We actually do this in an exercise in HA300. And if you want to really want to push this topic further, find out more about the Bulk File Loader option for mass flat file loading at high speed (covered in *HA350*).

Big Data Integration

Many organizations have embraced Big Data, collecting and storing staggering amounts of data of all types sourced from sensors, web logs, social media traffic, communication logs,

and more. Unlike traditional business data, Big Data is usually stored in an unstructured way and so it is difficult for us to define detailed semantics on this data so that it can easily be integrated with existing analytics. Organizations usually have to implement additional specialized tooling on top of their Big Data in order to add a semantic layer (to add meaning and structure to the data) and also provide query capabilities. These tools can be complex and are often used by only a small number of highly trained analysts. This often means that Big Data analysis becomes siloed from mainstream BI.

We accept that Big Data and enterprise data are typically stored separately. However, the analysis should not be separated, and business analysts should be able to consume any data and not be concerned over whether the data is Big or not.

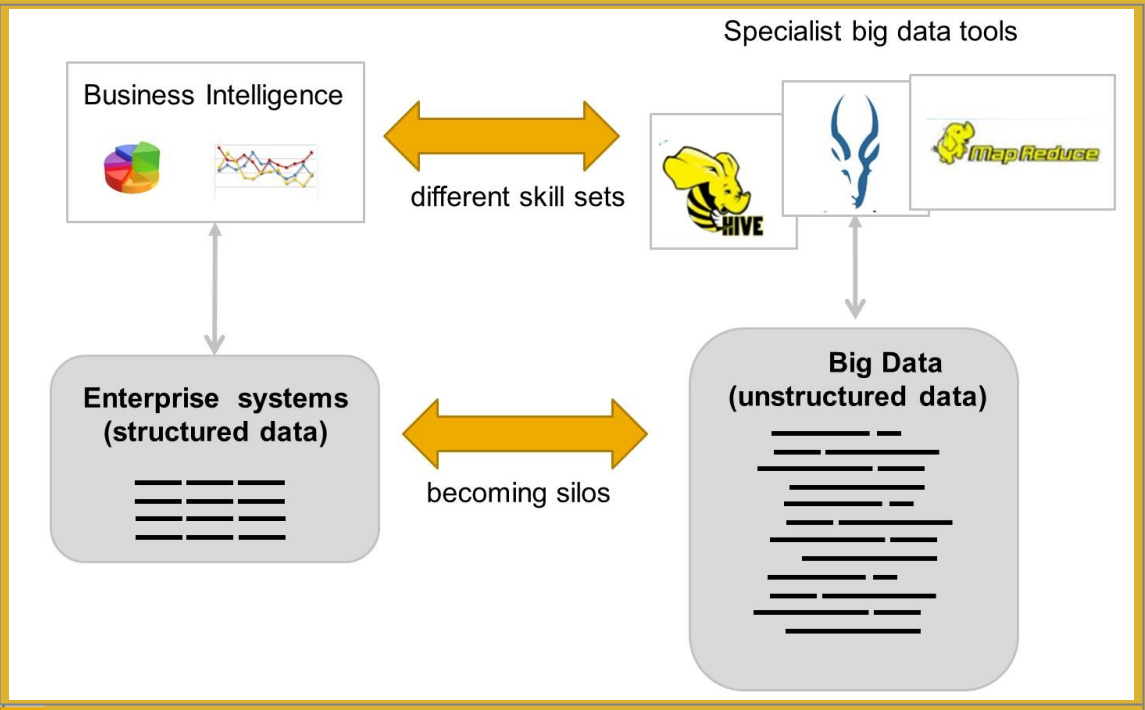


Figure 88: Big Data and Enterprise Data are Separate

SAP Vora enables analysts to consume Big Data and enterprise data as one, using their favorite drill-down, slice and dice, query tools.

Note:
SAP Vora was formerly known as SAP HANA Vora. SAP Vora has never depended on SAP HANA and can be used separately to integrate Big Data with any applications whether SAP HANA is in the landscape or not. The name change helps to remove this incorrectly assumed dependency with SAP HANA.

To describe SAP Vora from a technical perspective, first, a Big Data framework consists of a data storage component. The most popular Big Data storage solution is Apache Hadoop, usually referred to simply as Hadoop. It should be noted that there are other Big Data providers and SAP Vora will work with more and more providers including Microsoft Azure Data Lake.

Hadoop provides massive data storage capabilities across cheap, everyday servers that can easily be scaled to provide staggering storage capacity. However, Hadoop does not provide the data processing capabilities and that is where Apache Spark comes in.

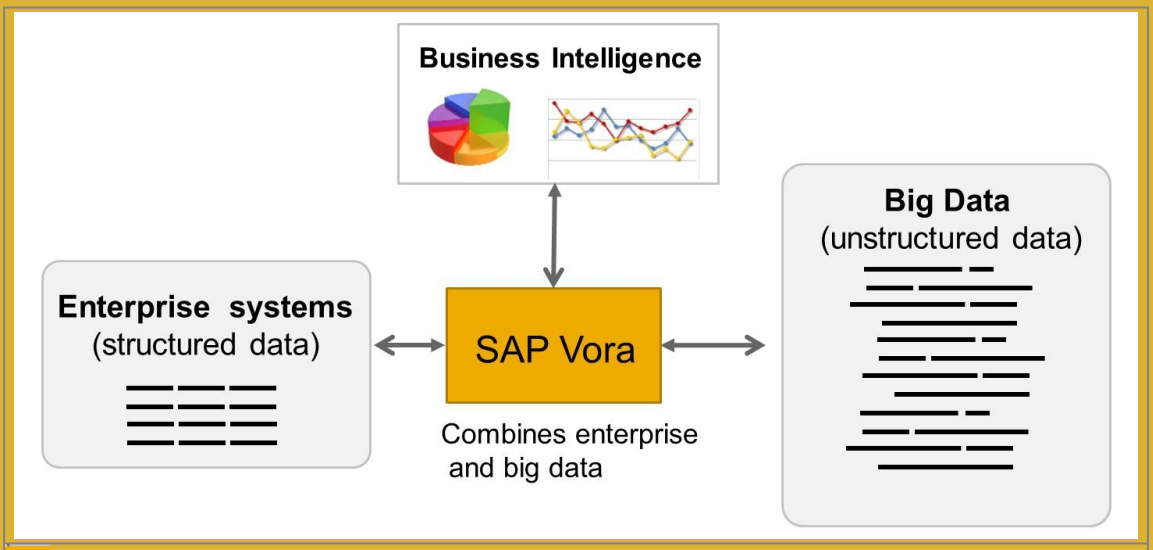


Figure 89: Bringing Big Data and Enterprise Data Together

Apache Spark (referred to simply as Spark), provides the query processing on top of Hadoop. However, Apache Spark is not able to provide the complex OLAP-type analysis features for Big Data data that most business analysts require. It is also not able to integrate enterprise data and Big Data. This is where SAP Vora comes in.

SAP Vora is an in-memory query engine that plugs into Apache Spark. It enhances Apache Spark to include hierarchies, including time-dependent hierarchies, OLAP-style dimensional drill-down/across, unit-of-measure conversion, currency conversion, and many more OLAP capabilities. SAP Vora also allows you to create precompiled queries that are ready to go, to enable fast execution.

The latest version of SAP Vora also includes support for other types of Big Data, not just Hadoop.

From an SAP HANA perspective, adding SAP Vora to the landscape brings Big Data analytics to an already powerful set of data modeling capabilities built using calculation views.

Using SAP HANA and SAP Vora you can perform the ultimate root-cause analysis by drilling down from the high-level business dashboard to the most atomic piece of individual data in a log. This provides a better understanding of the business context and helps to provide complete insight in one work flow.

Note:
When exploring SAP Vora you may come across **Altiscale**. Altiscale was one of the first companies to offer Hadoop as-a-service in the cloud, and was acquired by SAP in 2016. Altiscale is now a key part of SAP Cloud Platform Big Data Services.

The following are a list of possible SAP HANA Vora Use Cases:

1. **360-degree customer marketing** — Combine customer data with unstructured data from social media, e-mail, website activity, and discussion forums to better target sales and marketing.
2. **Cell phone service improvement** — Analyze instances of poor cellular service, such as dropped calls or poor audio, by drilling from billing data to detailed call log data.

3. **Fraud detection** — Detect anomalies and rogue trades by analyzing historical trends and current data simultaneously.
4. **Airline maintenance planning** — Combine aircraft sensor data, collected in-flight, with flight schedule and staffing data to optimally plan aircraft downtime.
5. **Targeted network maintenance and upgrades** — Analyze the impact of cable network congestion on churn, and identify which network upgrades will produce greater incremental revenue.

One of the key aims of SAP Vora is to support push-down of the query to the source of the Big Data. It is possible to use SDA to expose Big Data to SAP HANA directly, but this does not enable push-down of the query. In that case the data is extracted from Hadoop and presented to SAP HANA for processing. This does not encourage good performance.

LESSON SUMMARY

You should now be able to:

- Describe SAP HANA data acquisition solutions

SAP HANA Data Management Suite

195

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- SAP HANA Data Management Suite

SAP HANA Data Management Suite

Data Sprawl

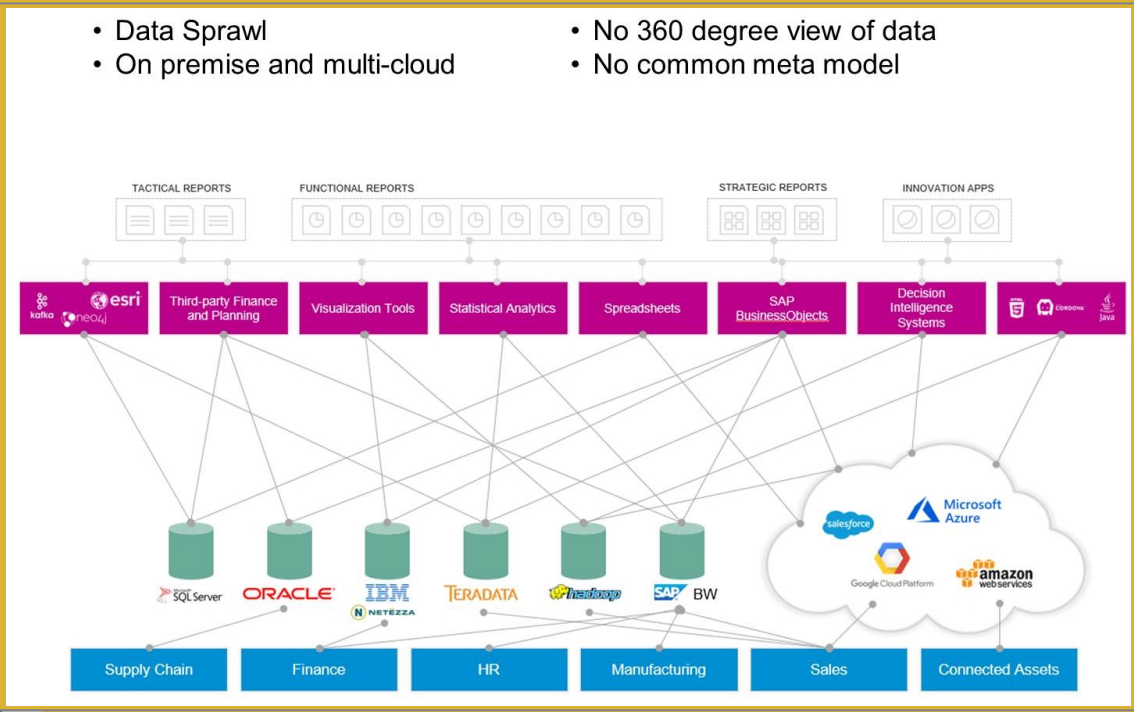


Figure 90: Data Sprawl

No-one would disagree that data is the new oil of the 21st century, and once data is refined and information is produced, this can unlock huge opportunities. However, data has become less accessible due to the proliferation of cloud based solutions and business units continue to build more applications further fragmenting the data landscape. Data becomes trapped in silos on-premise and in the cloud and this means that it is impossible to provide business decision makers with a 360 degree view of all information required to make good decisions. As data is become fragmented, a company's understanding of their customers, suppliers, and products has been in decline caused by data being inaccessible.

In addition, with data sprawl, there are substantial legal risks due to lack of data governance, for example, GDPR. Also, it is difficult to operationalize data science to use in everyday business processes when data cannot be harnessed.

SAP HANA Data Management Suite

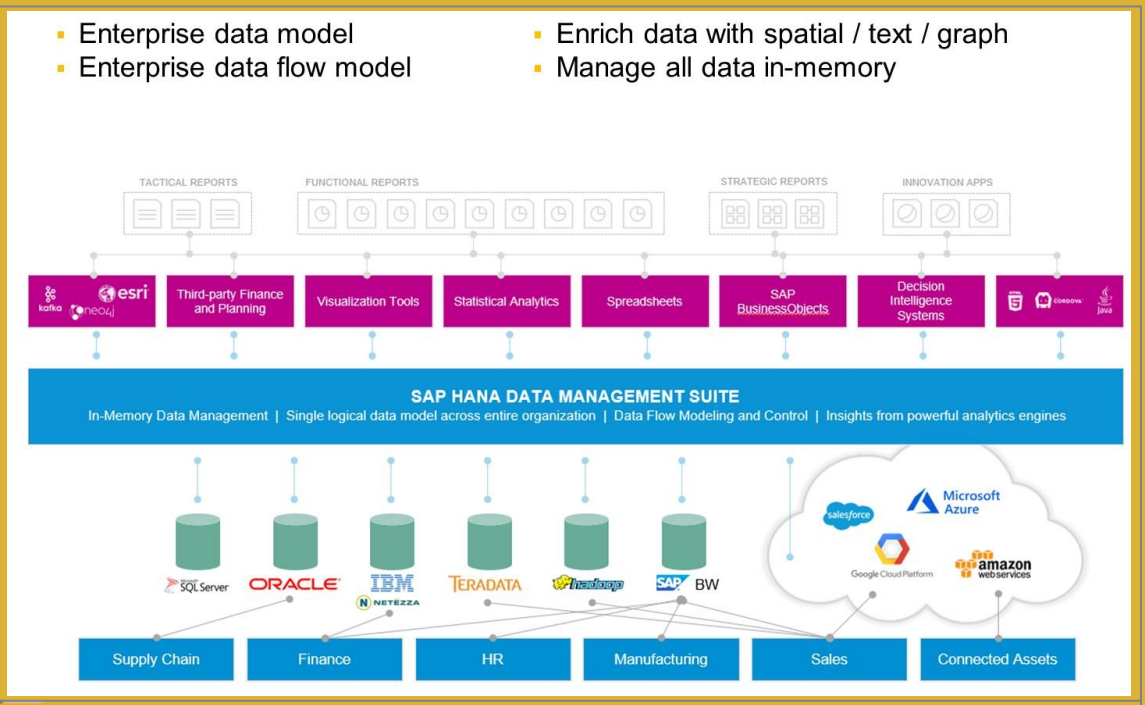


Figure 91: SAP HANA Data Management Suite

SAP HANA Data Management Suite provides the solution to the issues caused by data sprawl. It does this by providing the tools to define an enterprise-wide metadata model that identifies all sources of data regardless of where it is stored and harmonize the meaning of the data. HDMS The tools allow businesses to analyze the data lineage in order to trace the source of data all the way to reports.

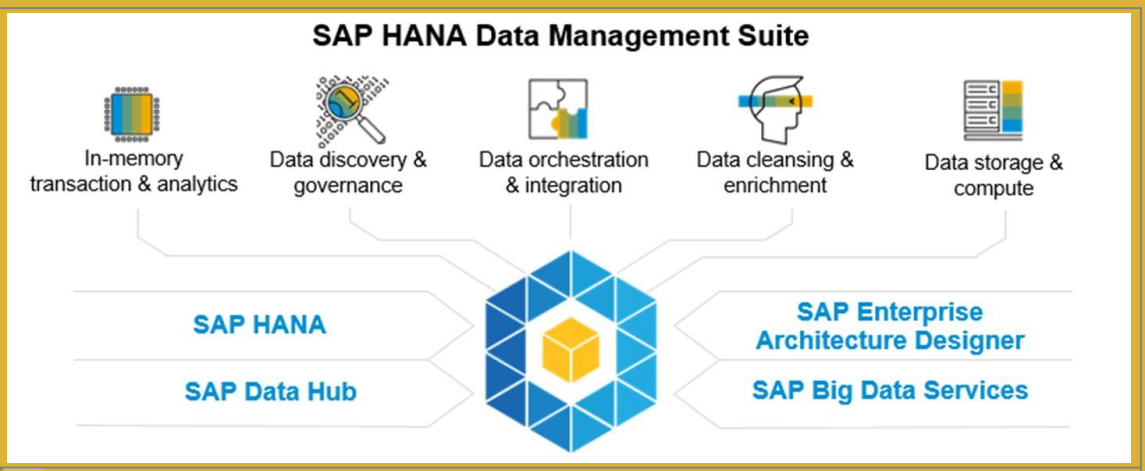


Figure 92: SAP HANA Data Management Suite

The suite includes a number of SAP products that are already available individually. These include:

- SAP HANA — provides the in-memory data processing services, ETL, virtualization, persistent memory technology, data anoymization services

- SAP Data Hub — provides centralized pipelining, orchestration, and data governance across federated data sources
- SAP Enterprise Architecture Designer — use to define manage the blueprint of data flow and data models and data rules across the enterprise
- SAP Big Data Services — fully managed Big Data services in the cloud.

SAP HANA Data Management Suite was a major announcement at SAPPHIRE June 2018 and will be the focus of development, so we can expect more capabilities in the near future

.

LESSON SUMMARY
You should now be able to:

- SAP HANA Data Management Suite

Learning Assessment

199

1. In SAP HANA, cold data is stored in HANA Extended Storage.

   Determine whether this statement is true or false.

   - [X] True
   - [X] False

2. What are typical characteristics of SAP HANA data replication?

   Choose the correct answers.

   - [X] A Virtualization
   - [X] B Duplicate data
   - [X] C Real-time
   - [X] D Continual streaming
   - [X] E Significant transformations

3. SAP SLT can be used for real-time replication from SAP and also non-SAP sources.

   Determine whether this statement is true or false.

   - [X] True
   - [X] False

4. What are features of SDA?

   Choose the correct answers.

   - [X] A Automatic data type translation
   - [X] B Data cleansing
   - [X] C Evaluation and execution of push-down possibilities to remote sources
   - [X] D Merging data from multiple sources

5. Which of the following are correct statements relating to the flat file import function of SAP HANA Studio?

   Choose the correct answers.

- [X] A The new table always has a 1:1 mapping between the file and table columns.
- [X] B The supported file types for upload are .csv, and .txt.
- [X] C When loading new data in a table that already contains data, the new data is appended to the existing data.
- [X] D Renaming of columns and changing data types is allowed when loading data to new HANA tables.

Learning Assessment - Answers

201

1. In SAP HANA, cold data is stored in HANA Extended Storage.

   Determine whether this statement is true or false.

   [X] True

   [X] False

   Correct!

2. What are typical characteristics of SAP HANA data replication?

   Choose the correct answers.

   [X] A Virtualization

   [X] B Duplicate data

   [X] C Real-time

   [X] D Continual streaming

   [X] E Significant transformations

   Correct!

3. SAP SLT can be used for real-time replication from SAP and also non-SAP sources.

   Determine whether this statement is true or false.

   [X] True

   [X] False

   Correct!

4. What are features of SDA?

   Choose the correct answers.

   - [X] A Automatic data type translation
   - [X] B Data cleansing
   - [X] C Evaluation and execution of push-down possibilities to remote sources
   - [X] D Merging data from multiple sources

   Correct!

5. Which of the following are correct statements relating to the flat file import function of SAP HANA Studio?

   Choose the correct answers.

   - [X] A The new table always has a 1:1 mapping between the file and table columns.
   - [X] B The supported file types for upload are .csv, and .txt.
   - [X] C When loading new data in a table that already contains data, the new data is appended to the existing data.
   - [X] D Renaming of columns and changing data types is allowed when loading data to new HANA tables.

   Correct!

UNIT 5 Developing and Running Applications on SAP HANA

UNIT OBJECTIVES

- Describe how enterprise suites run on SAP HANA
- Connect SAP Business Intelligence Tools to SAP HANA
- Outline how SAP BW leverages SAP HANA
- Describe SAP HANA Data Warehouse Foundation
- Describe the basics of native HANA applications

Unit 5
Lesson 1

Running SAP Enterprise Suites on SAP HANA

205

LESSON OBJECTIVES

After completing this lesson, you will be able to:

- Describe how enterprise suites run on SAP HANA

Running Business Suite on SAP HANA

The foundations of SAP Business Suite can be traced back to the early 1990s. Back then, the suite was called SAP R/3 and it consisted of a long list of application modules such as SD (sales and distribution), MM (materials management), and FI (finance) that supported key business processes. The application modules were powered by a technology platform called Basis, which provided all the underlying technology services needed to run the applications.

SAP R/3 was built to run on any of the leading enterprise relational database management systems (RDBMS) on the market. Back then, databases were disk-based, and tables were organized as row store. SAP R/3 was designed around the technology of that time, which meant many work-arounds were needed to maximize the performance of the system. For example, we needed to create separate aggregated tables to store summarized data, to help speed up management reporting. We also needed to build and maintain huge numbers of indexes to provide fast access to tables.

In the late 1990s, SAP R/3 was renamed ERP and was joined by many other key applications including CRM, SRM, and BW, to form a comprehensive suite of applications called SAP Business Suite. This significantly increased the range of business functions and processes available, and also provided solutions to connect customers and business partners in an emerging internet based world.

SAP Business Suite ran on a technology platform called SAP NetWeaver, the core of which is still the well-established Basis. However, SAP NetWeaver offered significantly more technology services than Basis, including data integration tools, and it also provided a modern web-services-based development infrastructure that supported service-oriented architecture (SOA).

The key principle of NetWeaver was to provide an enterprise-wide technology **platform** to not only power SAP Business Suite, but to support custom application development with all the tools needed in the SAP NetWeaver toolbox.

However, the underlying application design of SAP Business Suite and also NetWeaver services was still based around disk-based, row storage and also the hardware technology of that time. The work-arounds to help improve performance were still needed. In fact as time went on, the code and data model became increasingly complex. To help with performance, we moved data completely out of the applications and copied it each night to a dedicated storage component. This eased the pressure from the business transactional system and also enabled better performance for reporting users who were not competing for resources with the transactional users. However, the downside is that the IT landscape had become complex and data was being duplicated.

Business Suite Powered by SAP HANA Benefits

SAP HANA is a full relational database. It can be used wherever a relational database is needed to power any type of application. This includes SAP Business Suite. SAP wasted no time in making SAP HANA available as a database to run SAP Business Suite, so that customers could move away from the old disk-based, row store technology to a modern platform.

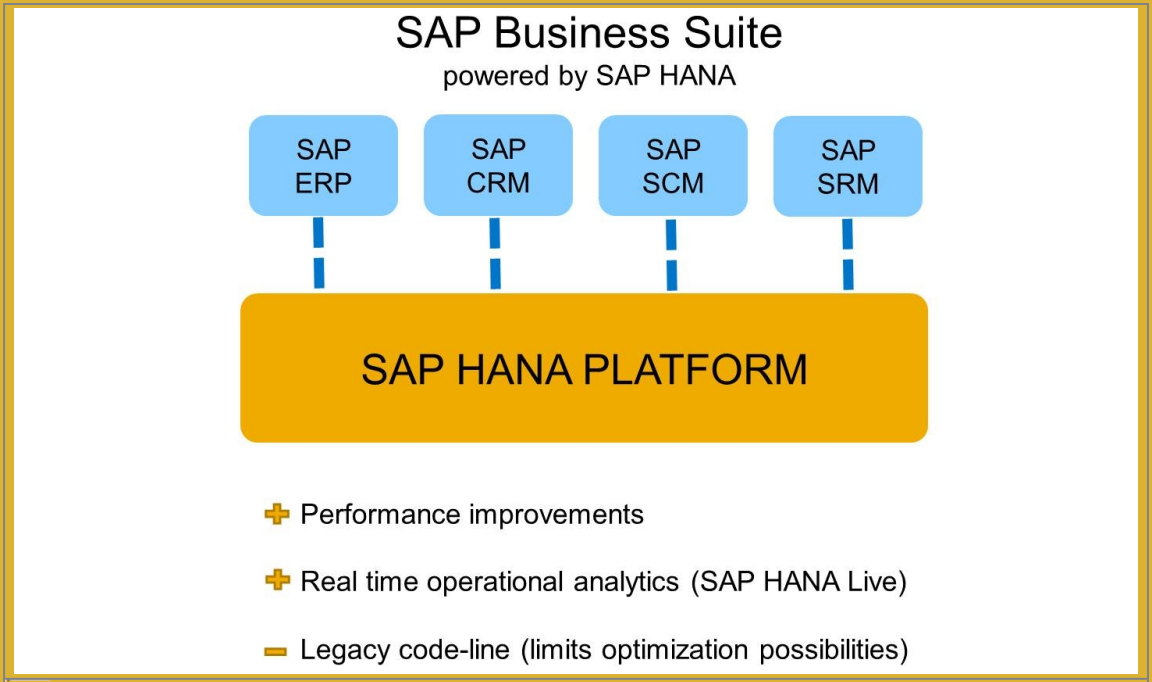


Figure 93: BS powered by HANA

SAP Business Suite powered by SAP HANA was born and provided immediate benefits. The following list outlines a few of those benefits:

- Massive speed-up of performance of existing applications
- A possibility to simplify the IT landscape by combining transactions and operational analytics back into one system
- Real-time reporting on operational data
- A new interface called SAP Fiori to modernize the user experience
- A platform that not only runs SAP applications, but provides a digital platform to power partner and customer developed applications

However, remember that the core code-line for SAP Business Suite was developed many years ago and its design was based on the technology at that time (disk based, limited memory, and so on). This means that we are not able to take advantage of potential optimizations and simplifications that come from SAP HANA. Also, bear in mind that SAP Business Suite code-line still has to be compatible with non-HANA databases, and that also limits what we can do with the code.

Today, large numbers of SAP customers have migrated the legacy databases that run their SAP Business Suite, so they now run on SAP HANA. SAP provides all the tools to make the migration easy and, mostly, automated.

> Note:
> Despite it being commonly used, SAP Business Suite on SAP HANA is not the official name. It is SAP Business Suite powered by SAP HANA.

SAP Business Suite still requires SAP NetWeaver to provide the underlying technology layer on the application side. SAP Business Suite is built using ABAP, and NetWeaver is needed to provide the ABAP development and runtime environment. However, with SAP Business Suite on SAP HANA, SAP have optimized the ABAP code to run on SAP HANA, to ensure that the best performance is possible. The data models remain the same, which means that the familiar ABAP table names are not changed, and custom developments continue to work with little or no adjustment.

To a business user, the migration is non-disruptive. There are new SAP Fiori interfaces available, but these are optional and users can continue to work with SAPGUI as before. The biggest change for the user is the massive improvement in performance of applications, especially batch processes such as MRP and period end preparation and closing. They also benefit from access to real-time reports and dashboards on operational data.

SAP S/4HANA Runs on SAP HANA

What is SAP S/4HANA?

SAP S/4HANA (Suite for HANA) is SAP's next-generation business suite for the digital world. SAP S/4HANA is built and optimized to run only on the SAP HANA database. It is not available on any other third party database such as Oracle, IBM, or Microsoft.

SAP S/4HANA includes solutions for all key business processes such as procure-to-pay, order–to-cash, and so on.

Many customers have migrated from Business Suite on anyDB to Business Suite powered by SAP HANA to obtain a massive speed-up in performance of their business applications. The move to SAP S/4HANA is often the fist major step for a customer who is building a next-generation business platform.

For customers who did not migrate from Business Suite on anyDB to Business Suite powered by SAP HANA, they can upgrade and migrate straight to SAP S/4HANA. Customers running non-SAP enterprise suites can implement SAP S/4HANA and take advantage of the supplied tools to simplify and shorten the data migration effort.

SAP S/4HANA was completely rewritten from scratch. Just like Business Suite, SAP S/4HANA is still written in ABAP, but the ABAP code was rewritten to exploit the full power of the SAP HANA in-memory processing engines and database. By contrast, Business Suite on SAP HANA uses the same ABAP code as Business Suite on anyDB, but the code was adjusted to ensure that it could run on SAP HANA with some optimizations added.

A rewrite for SAP S/4HANA was needed because the code line for Business Suite was written a long time ago and the coding approach fitted the technology of that time. There were many approaches used to ensure best performance on legacy databases that are simply not needed today with SAP HANA. For example, SAP S/4HANA does not need to store aggregated data. The SAP HANA database used by SAP S/4HANA can summarize data on the fly. As well as this, SAP S/4HANA does not need indexes, as the design of the column store database means that we already have fast access to any combination of data columns.

Business Suite has to support multiple databases, so the code became complex to ensure compatibility with all of them. SAP S/4HANA only runs on SAP HANA, and so we could drop

the unnecessary code and produce a leaner, faster, more efficient ERP suite that is ready to move companies forward with a foundation for their digital transformation.

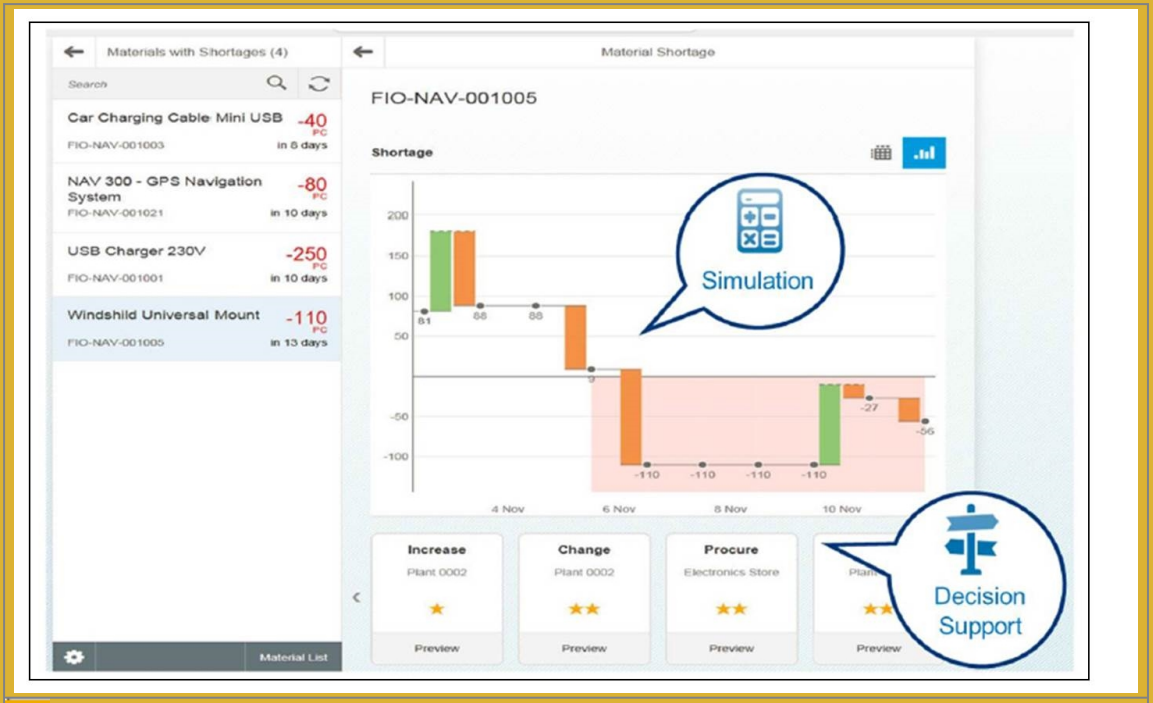


Figure 94: SAP S/4HANA

Like Business Suite on SAP HANA, SAP S/4HANA provides a new interface called SAP Fiori, which improves performance and user experience. SAP Fiori applications are intuitive and simple to use, with a modern look and feel. SAP Fiori applications run on any device. Unlike SAPGUI, users are no longer tied to their desktops, so they can switch devices and work with the same applications.

A major difference between Business Suite and SAP S/4HANA is the provision of many brand new applications. Business Suite powered by SAP HANA focuses on continuity. SAP S/4HANA provides continuity, while also providing next-generation digital applications that combine in-memory analytics and transactional processing. SAP have rebuilt many traditional applications so that they provide the business user with analytics, right in place where decisions are needed.

Embedded Analytics

A key component delivered with SAP S/4HANA is Embedded Analytics.

Embedded Analytics provides a complete enterprise-wide data model, built using ABAP Core Data Services (CDS). The data model delivers thousands of consumption-ready CDS views of real-time business data. SAP S/4HANA Embedded Analytics also includes a built-in analytic engine (based on BW) to handle complex OLAP requests in memory. Also supplied are built-in, easy-to-use query building tools. It is possible to use any existing SAP reporting tool, such as Analysis, Web Intelligence or Lumira with Embedded Analytics.

It might sound surprising, but many students confuse SAP S/4HANA with SAP HANA and believe them to be the same thing. So one of the aims of this lesson is to help make that distinction by emphasizing the role of SAP HANA in S/4HANA.

LESSON SUMMARY

You should now be able to:

- Describe how enterprise suites run on SAP HANA

Unit 5
Lesson 2

209

Connecting SAP Business Intelligence Tools to SAP HANA

LESSON OVERVIEW
This lesson covers connecting Business Intelligence (BI) tools to SAP HANA.

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Connect SAP Business Intelligence Tools to SAP HANA

Connecting SAP BI to SAP HANA

A popular use case for SAP HANA is to power real-time business intelligence (BI).

SAP HANA can be deployed as a local data mart, to capture data from various operational systems, including SAP enterprise systems and non-SAP sources. Data can be loaded to SAP HANA either using regular batch loading or real-time updates. It is even possible to have SAP HANA connect to data sources in remote systems directly so that data loading to SAP HANA is not necessary. SAP HANA then become a logical layer sitting between the data sources and the reporting tools to provide the data model.

SAP HANA uses many industry standard connection protocols to connect to the database. SAP HANA can connect directly to a large number of reporting tools, including SAP Business Objects. So, reporting on SAP HANA can begin immediately.

For customers who run SAP enterprise suites such as Business Suite or SAP S/4HANA, one of the most appealing aspects of the SAP HANA BI use case is that SAP HANA comes with extensive built-in virtual data models. These models provide real-time business-ready views of all SAP enterprise operational data, based on the tables of either SAP Business Suite or SAP S/4HANA. SAP have developed and maintained these comprehensive virtual data models to expose live operational tables from all areas of SAP Business Suite and SAP S/4HANA.

The real value in the virtual data models is the business semantics that are provided by SAP. Raw database tables are combined, filters, and calculations are added to expose business views ready for immediate consumption by any reporting tool with no additional modeling needed. So, instead of having to define the data models and semantics in your reporting tool, this is be done once in SAP HANA and shared by all tools.

As well as consuming the ready-made data models provided by SAP — either calculation views for Business Suite, or ABAP CDS for S/4HANA — you can create your own custom calculation views and consume these from any reporting tool.

Connecting Reporting Tools

Because SAP HANA provides database access using industry standard connectors, all SAP reporting tools are able to connect to SAP HANA directly using one or more of the following:

- ODBC — for generic relational connections

- JDBC — for JAVA-based relational connections
- ODBO — for multidimensional data source connections
- BICS — SAP proprietary connector only used by SAP reporting tools

Some reporting tools offer a choice of connectors. For example, with Crystal Reports you can use either a JDBC or ODBC connectors. Some connectors support more SAP HANA features.

The query language used depends on the reporting tool used and the connection type. The two database query languages used by SAP HANA are SQL (for relational models) and MDX (for multidimensional models).

SAP BI tools

Let's describe the current, recommended SAP BI tools:

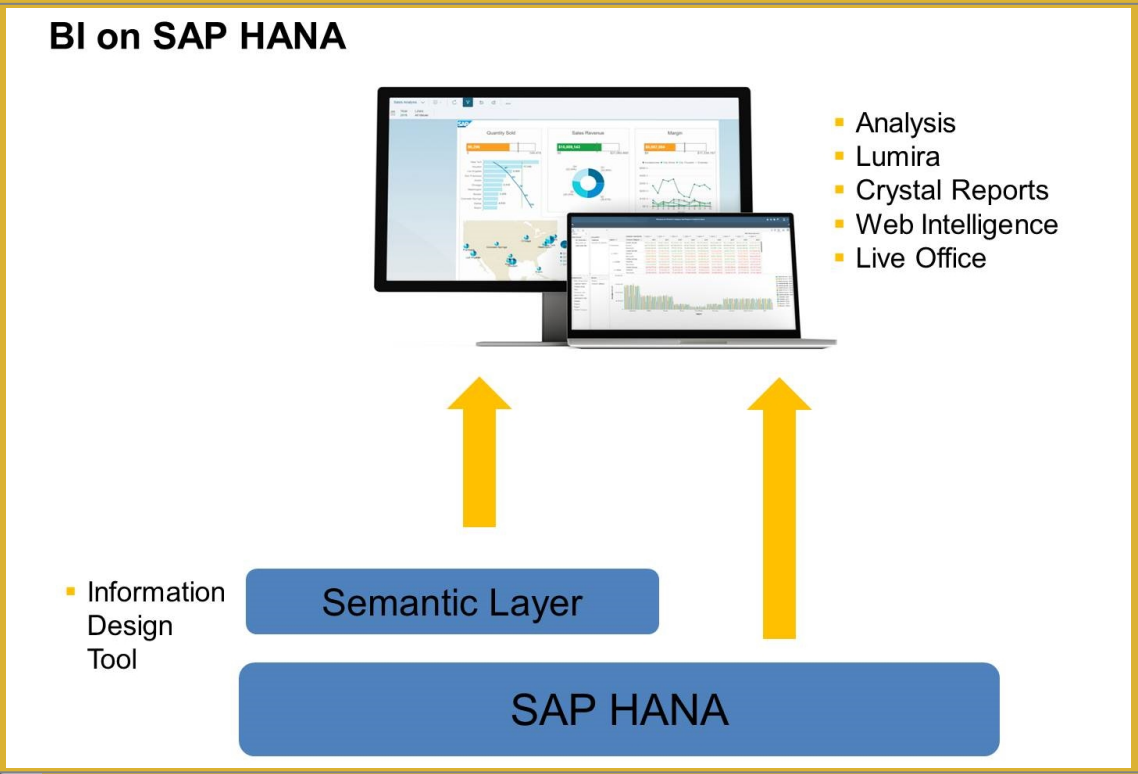


Figure 95: On Premise Reporting Suite

- **SAP Analysis, Edition for Microsoft Office** — Powerful OLAP reporting in a well known and popular Excel environment
- **Lumira** — Build intelligent dashboard and cockpits that can be deployed automatically to mobile devices
- **Web Intelligence** — End user, self service query tool using a simple wysiwyg drag and drop interface.
- **Crystal** — Create consumer grade, professional looking enterprise reports that require precise layout and formatting.
- **Live Office** — Embed real-time Business Objects reporting content into Microsoft Office documents.
-

These tools can be implemented standalone or as part of an implementation of the powerful and popular **Business Objects Platform**, where user administration, security and report storage and access can be centrally managed. With the Business Objects Platform, reports can also be scheduled. Using the bursting capabilities, report content can be broken up and distributed to various user groups.

Semantic Layer

All SAP reporting tools can connect to SAP HANA directly, however is it also possible to build an additional semantic layer into the BI stack.

Using the **Information Design Tool (IDT)** it is possible to create a sophisticated data model that combines multiple data sources (including SAP HANA) into a harmonized model. This is typically not needed where data integration is managed by SAP HANA. However, if data is integrated at a layer above SAP HANA this approach might make sense.

Note:
The IDT tool is an evolution of the Business Objects Universe Designer. The Semantic Layer is the successor to the Universe.

Unit 5
Exercise 9

213

Consume a Calculation View with SAP Analysis for Microsoft Excel

Exercise Objectives

In this exercise, learn how to build a report that consumes a Calculation view of type CUBE.

Business Example

You have been tasked with creating a flexible report in Analysis for Microsoft Excel on top of a HANA Calculation view. The report is to be used to analyze sales to US customers, with the possibility to drill-down to examine sales by any combination of customer and product attributes.

> Note:
> In this exercise, when values include ##, replace these characters with your own student number.

1. Start *Analysis for Microsoft Excel* and open a blank workbook.
2. Insert a new data source based on the SAP HANA calculation view *US Product Sales* which is found in the folder *HA100_00_HDI_CONTAINER_1* → *HA100*. Use the log on details from the table, *Logon Details*. Make sure cell *A1* is selected before you begin.

   Table 13: Logon Details

Field	Value
Connection	*HTTP_HANA*
User	**STUDENT##**
Password	**Training1**

3. Display the *Quantity* and *Amount* by *Delivery* attribute.
4. Break down the data by *Product Group*.
5. Remove the *Delivery* attribute and now add the *Location* in the columns to create a cross tab report.
6. Close *Analysis for Microsoft Excel*.

Note:
In case you are wondering why you were not able to select the calculation view *US Product Sales* that you created in the previous exercise, here is an explanation. In SAP HANA 2.0, where we use container based development, database objects (for example, tables, views, functions) are not directly accessible to an application user such as a reporting user. Access to database objects is only allowed by an internal technical HANA user and this technical user is generated automatically when a container is first built. This technical user is the real database user. The application user sits outside the database. But sometimes we need to allow an application user to also have access to the database objects whenever the application directly consumes a database object. An example of this scenario is when an *Analysis for Microsoft Excel* user tries to select a calculation view from the HANA database to include as a data source. By default they are not able to see any calculation views as the database doesn't offer its objects to this application user. So what we need to do is explicitly grant permissions to the application user (similar to the ones the internal technical user already has) and then the calculation view you created earlier would then appear for selection. We decided for this HA100 overview course to avoid these additional steps of granting permissions to keep things simple. But these steps are covered in the HA300 course.

Consume a Calculation View with SAP Analysis for Microsoft Excel

215

Exercise Objectives

In this exercise, learn how to build a report that consumes a Calculation view of type CUBE.

Business Example

You have been tasked with creating a flexible report in Analysis for Microsoft Excel on top of a HANA Calculation view. The report is to be used to analyze sales to US customers, with the possibility to drill-down to examine sales by any combination of customer and product attributes.

> Note:
> In this exercise, when values include ##, replace these characters with your own student number.

1. Start *Analysis for Microsoft Excel* and open a blank workbook.
   a) In the Windows Start page, enter **`Analysis`**, and in the search results on the right of the screen, choose *Analysis for Microsoft Excel*.
   b) At the next prompt, double-click *Blank workbook*.
2. Insert a new data source based on the SAP HANA calculation view *US Product Sales* which is found in the folder *HA100_00_HDI_CONTAINER_1* → *HA100*. Use the log on details from the table, *Logon Details*. Make sure cell *A1* is selected before you begin.

   Table 13: Logon Details

Field	Value
Connection	*HTTP_HANA*
User	**`STUDENT##`**
Password	**`Training1`**

   a) Make sure that cell *A1* is selected.
   b) From the Excel toolbar, choose the *Analysis* tab.
   c) On the *Analysis* ribbon, choose *Insert Data Source*.
   d) From the menu, choose *Select Data Source*.

e) In the next dialog, choose *Skip* to avoid logging on to HANA database using the SAP BusinessObjects Platform.

f) In the next dialog, choose the connection from the table, *Logon Details*, and choose *Next*.

g) In the next dialog, enter the user and password provided in the table, *Logon Details*.

h) In the *Select Data Source* dialog box, choose the *Area* tab.

i) Expand the folder *HA100_00_HDI_CONTAINER_1* → *HA100* and highlight the calculation view *US Product Sales*.

j) Choose *OK*.

3. Display the *Quantity* and *Amount* by *Delivery* attribute.
   a) Drag the *Delivery* attribute from the *Data Source* pane to the rows pane.
   The measures are now split by the two types of Delivery: *CHARGEABLE* and *FOC*.

4. Break down the data by *Product Group*.
   a) Drag the *PRODUCT_GROUP_TEXT* attribute from the *Data Source* pane to the rows pane.
   The measures are now split by *Delivery* and also *Product Group.*

5. Remove the *Delivery* attribute and now add the *Location* in the columns to create a cross tab report.
   a) Drag *Delivery* from the rows to the *Data Source* pane to remove it from the results (or right-click and choose *Remove)*

   b) Drag *Location* from the *Data Source* pane to the columns pane.

6. Close *Analysis for Microsoft Excel*.
   a) Close *Analysis for Microsoft Excel*.

Note:
In case you are wondering why you were not able to select the calculation view *US Product Sales* that you created in the previous exercise, here is an explanation. In SAP HANA 2.0, where we use container based development, database objects (for example, tables, views, functions) are not directly accessible to an application user such as a reporting user. Access to database objects is only allowed by an internal technical HANA user and this technical user is generated automatically when a container is first built. This technical user is the real database user. The application user sits outside the database. But sometimes we need to allow an application user to also have access to the database objects whenever the application directly consumes a database object. An example of this scenario is when an *Analysis for Microsoft Excel* user tries to select a calculation view from the HANA database to include as a data source. By default they are not able to see any calculation views as the database doesn't offer its objects to this application user. So what we need to do is explicitly grant permissions to the application user (similar to the ones the internal technical user already has) and then the calculation view you created earlier would then appear for selection. We decided for this HA100 overview course to avoid these additional steps of granting permissions to keep things simple. But these steps are covered in the HA300 course.

SAP Analytics Cloud and SAP HANA

SAP Analytics Cloud (SAC)

Increasing numbers of organizations are moving their analytics to the cloud. SAP Analytics Cloud (SAC) provides a comprehensive tool set for developing and deploying BI content that is easily shared.

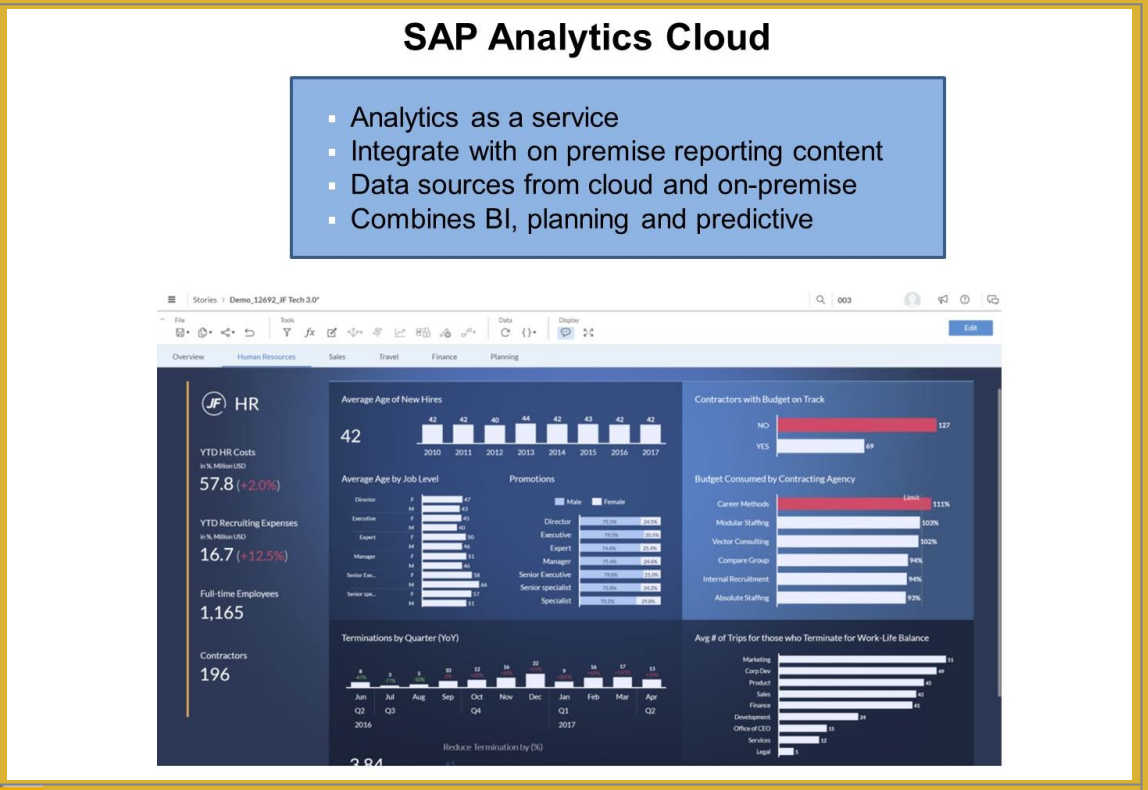


Figure 96: SAP Analytics Cloud

To support SAP's analytics strategy of **hybrid analytics** (on cloud mixed with on-premise analytics), SAC can take live data sources from on-premise (such as Universes, BW and SAP HANA) and also cloud data. SAC also combines reporting content from on premise tools such as Web Intelligence and combine with cloud developments.

SAC provides a full suite combining BI, planning, and predictive capabilities and plenty of industry content is already available for quick start.

The SAP Analytics Cloud Digital Boardroom provides C level executives with a 360 degree views of all KPIs with the ability to drill down and explore the underlying data.

LESSON SUMMARY

You should now be able to:

- Connect SAP Business Intelligence Tools to SAP HANA

Unit 5
Lesson 3

219

Outlining SAP Business Warehouse powered by SAP HANA

LESSON OVERVIEW
This lesson covers SAP Business Warehouse (BW) on SAP HANA.

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Outline how SAP BW leverages SAP HANA

SAP Business Warehouse (BW) powered by SAP HANA

SAP Business Warehouse (BW) was the first SAP application enabled to run on SAP HANA. In recent years, large numbers of customers have been migrating their disk-based legacy databases under BW, to SAP HANA. This immediately provides them with superior performance from their SAP BW solutions, particularly in the key areas of reporting and data loading. It also opens up powerful data modeling possibilities by combining BW and core HANA modeling techniques.

Note:
SAP HANA was developed using a lot of the technology first developed for Business Warehouse Accelerator (BWA), the in-memory appliance that improves read performance of InfoCubes.

SAP BW Simplified when Running on SAP HANA

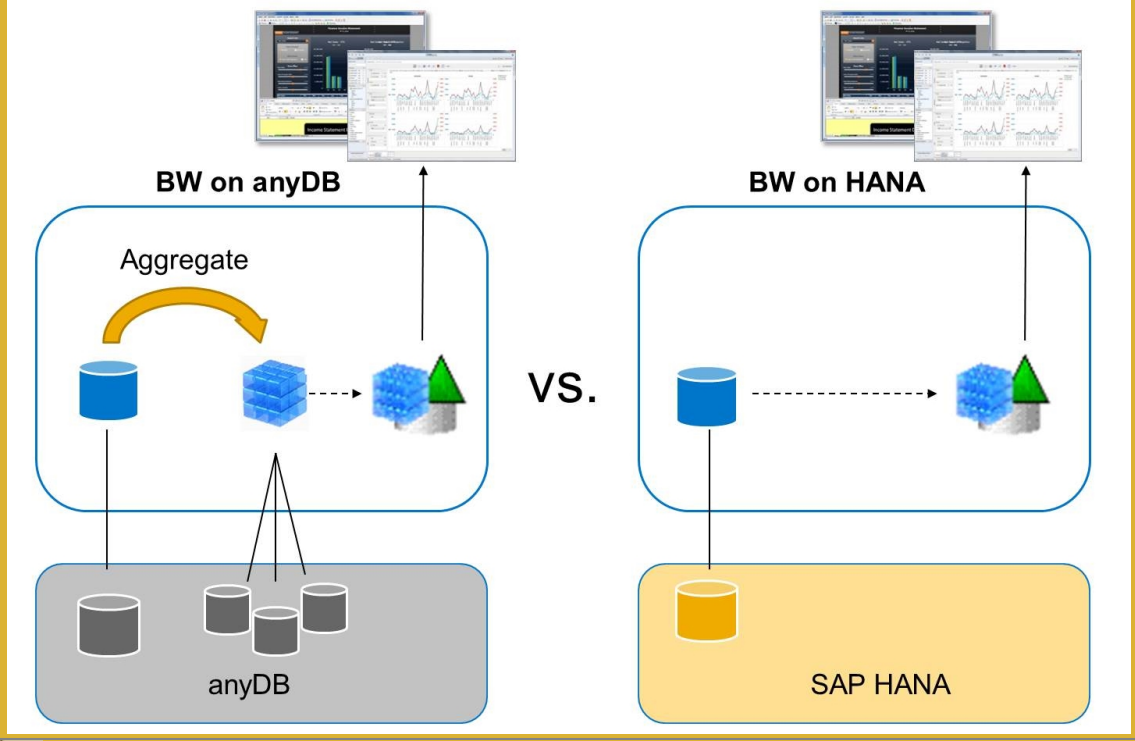


Figure 97: SAP BW Simplified when Running on SAP HANA

With SAP BW powered by SAP HANA, it is not just the speed that improves dramatically. Other major benefits include a significantly reduced data footprint.

A non-HANA SAP BW design involves building many layers of stored aggregations. We do this in order to improve the speed of reporting by precalculating business data in advance of the users requesting it. With the raw power of SAP HANA, aggregations are done on the fly when the user runs a query, so you no longer need to build and store aggregates.

By removing the aggregates, which are usually modeled using BW InfoCubes, you not only reduce the data footprint, but also dramatically simplify the data modeling tasks. The data loading is quicker when you do not need additional layers. Plus, it is easier to implement new flows and integrate them into existing models when you only need to manage one layer of data and not have to consider layers of aggregations.

With SAP BW powered by SAP HANA, the only persistent layer is modeled with Advanced DataStore Objects (ADSO). These are only available with SAP BW powered by SAP HANA and are optimized for in-memory processing.

Integrate any Data with BW

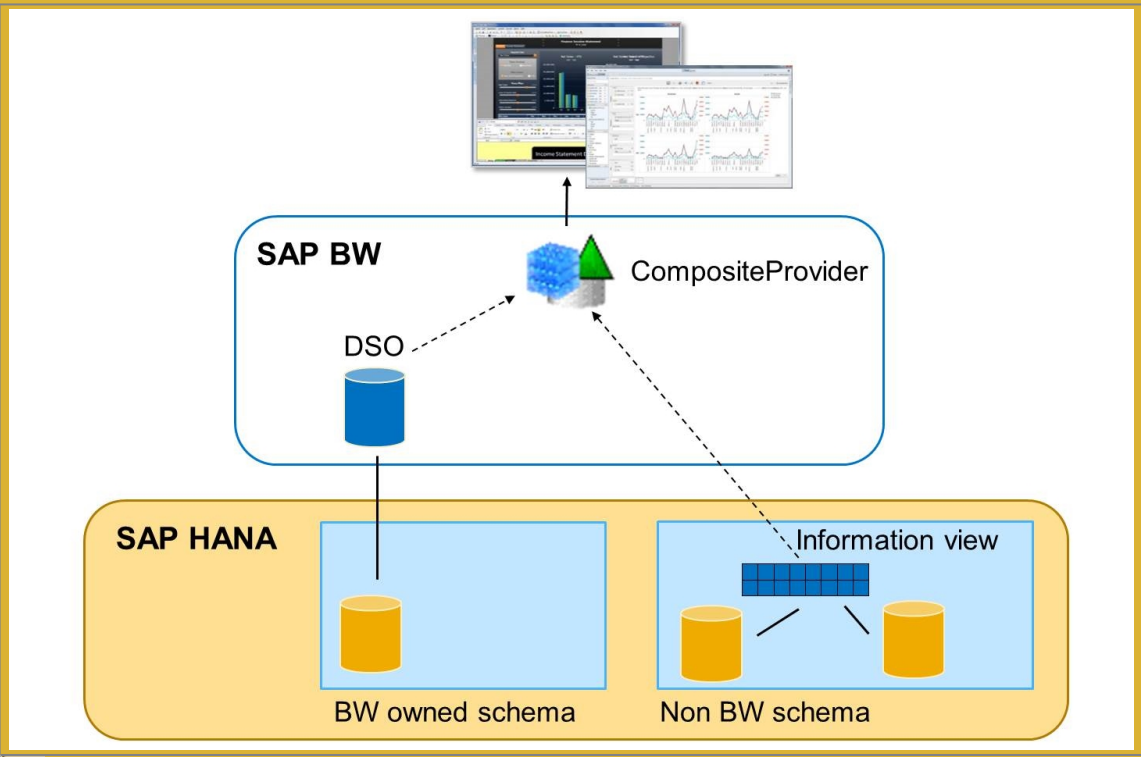


Figure 98: Integrate any Data with SAP BW

When running SAP BW powered by SAP HANA, you have access to all external SAP HANA schemas. You can leverage the advanced data acquisition mechanisms that are part of SAP HANA, such as SDA, SDI, SLT, and streaming, to provision data to these external schemas. You can then build SAP HANA Calculation views over the tables in the external schemas and integrate these views with SAP BW modeled objects, such as CompositeProviders.

This means that you can combine the data warehouse data, perhaps representing the historical data, with SAP HANA data in external schemas representing live operational data. SAP BW reports and applications can then consume the combined data, giving you a complete picture of historical and up-to-date data for your real-time applications. This combination of the modeling features of SAP BW and SAP HANA is called mixed modeling.

Integrate SAP BW Data with any Application

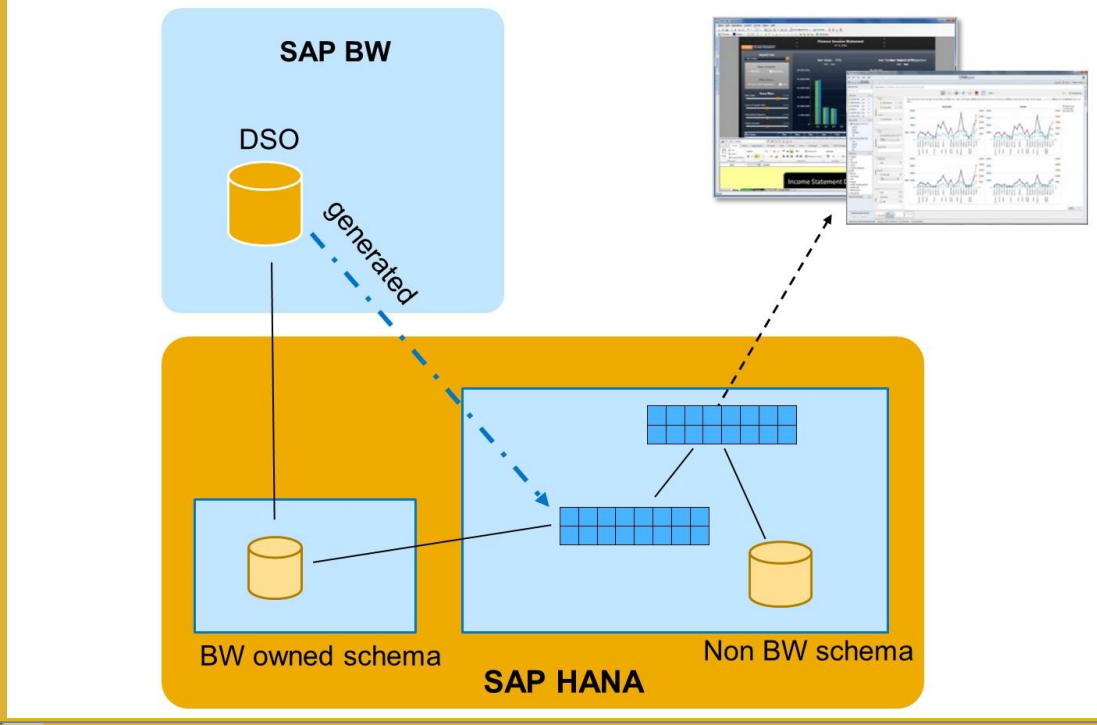


Figure 99: Integrate SAP BW Data with any Application

SAP BW is able to automatically generate SAP HANA calculation views that correspond to InfoCubes, DataStore Objects, and other SAP BW modeled objects. You can then integrate these generated SAP HANA calculation views into other SAP HANA views, which could provide additional calculations, to create a mash-up that can be consumed by any tools (including re-consuming by SAP BW).

This means that you can expose SAP BW data to any application that can connect to SAP HANA. If you change the design of SAP BW objects, the corresponding SAP HANA calculation views are automatically regenerated so everything is kept in step.

SAP BW Powered by SAP HANA Versus SAP BW/4HANA

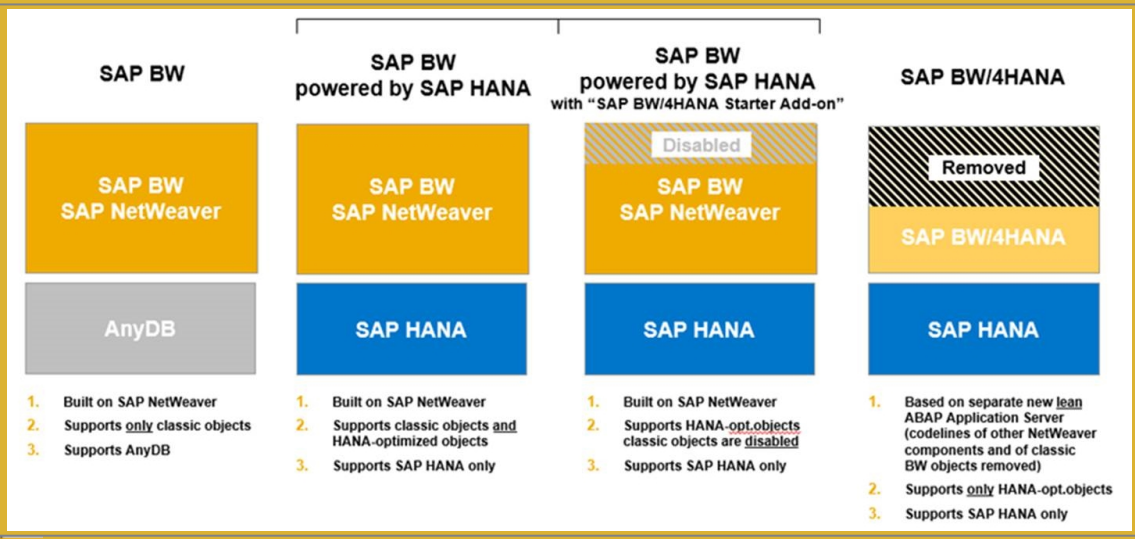


Figure 100: SAP BW Powered by SAP HANA Versus SAP BW/4HANA

There are two versions of SAP BW that can run on SAP HANA. They are as follows:

- SAP BW powered by SAP HANA

  This is supported for SAP BW 7.4 and SAP BW 7.5, and is popular with customers who are transitioning from SAP BW on anyDB. With this edition, it is possible to create and maintain the legacy BW objects, such as InfoCubes, alongside the new HANA optimized objects, such as Advanced DSO and CompositeProviders.

  Modeling, query building, and administration is carried out using the traditional SAPGUI interfaces. Eclipse can also be used for some of this. Both 3.x and 7.x data flows are supported in this version.

  A few years after introducing SAP BW powered by SAP HANA, SAP developed an add-in that can be installed. Its purpose is to shut down the classic objects of SAP BW. Customers do not have to install the add-in, but if they do, it eliminates the risk of BW objects being created that are not optimized for HANA. This is often done as part of the preparation for the move to SAP BW/4HANA, where classic objects are not supported. The last version of SAP BW powered by SAP HANA was 7.5.

- SAP BW/4HANA

  This is a brand new BW written from scratch. It represents the next-generation SAP data warehouse. Although it is built on the principles of classic BW, this edition supports only SAP HANA-optimized objects. Therefore, huge amounts of ABAP code, and features that were not needed, can be removed, to create much leaner BW.

  With this version it is only possible to create and maintain SAP HANA-optimized BW modeling objects, such as CompositeProviders. InfoCubes, MultiProviders are no longer supported.

  BEx Query Designer is not used. Queries are built using the Eclipse tools and SAPGUI is not needed for modeling. Customers can upgrade from BW powered by SAP HANA to SAP BW/4HANA using supplied tools. However, customers must first remove or migrate all classic objects to the new SAP HANA-optimized objects. SAP BW/4HANA is a new product and so has a new version numbering that started at 1.0.

You can compare BW powered by SAP HANA to Business Suite powered by SAP HANA, and SAP BW/4HANA to SAP S/4HANA. The former are code adjustments and maintain support for non-SAP HANA databases. The latter are complete code rewrites and support only SAP HANA.

Note:
SAP BW powered by SAP HANA is often referred to as SAP BW on HANA, but this is not the official title. Also, SAP BW/4HANA is still new to the market and is often confused with SAP BW powered by SAP HANA. Make sure you don't confuse these.

LESSON SUMMARY
You should now be able to:

- Outline how SAP BW leverages SAP HANA

Unit 5
Lesson 4

Developing SQL Data Warehouses with SAP HANA

225

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Describe SAP HANA Data Warehouse Foundation

Building a Custom Data Warehouse with SAP HANA

Remember to emphasize that an SAP-HANA-based data warehouse is implemented by companies who prefer to build their data warehouse from the ground up rather than purchase a complete, out of the box solution (SAP BW). SAP BW is still SAP's enterprise data warehousing solution and is now more powerful than ever since it is powered by HANA. However, BW consultants are advised to learn more about this 'build it yourself' approach, especially as customers will demand to know what is the difference in the two approaches. BW consultants already possess many of the fundamental skills and data warehousing knowledge required to get started with the native SAP HANA approach.

For many people, when they hear references to SAP data warehousing, they immediately think of SAP BW powered by SAP HANA, and more recently, SAP BW/4HANA. Both of these remain SAP's application (out of the box) driven approach for enterprise-wide data warehousing. However, SAP HANA offers an alternative approach, which does not require SAP BW or SAP BW/4HANA.

All tools required to build a high-performance SQL data warehouse are provided with the SAP HANA platform. This is referred to as the **Native SQL driven approach with SAP HANA**

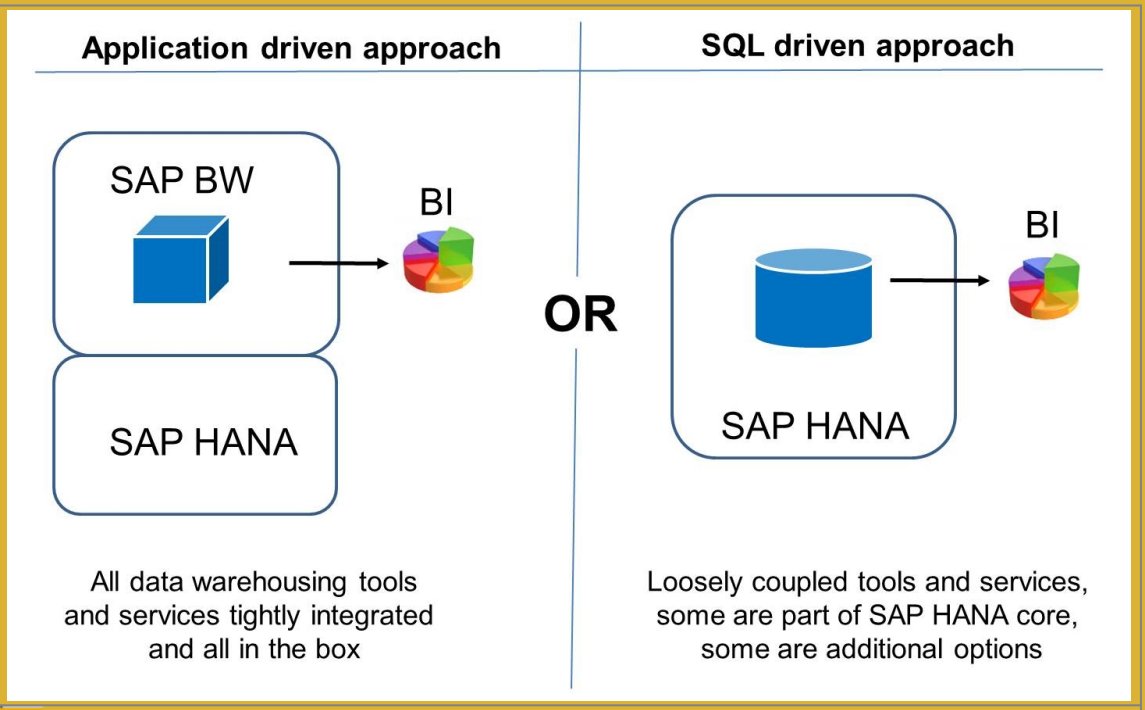


Figure 101: Choosing a Data Warehousing Approach

With the application-driven approach, all tooling and services, extractors, data governance, and modeling objects needed to support a data warehouse are already built by SAP and are ready to go. Another aspect to consider is that SAP BW and SAP BW/4HANA are tightly integrated in to SAP enterprise suites and many other SAP applications, and that makes them appealing to customers who have already implemented SAP applications.

A data warehouse built using only SAP HANA is called an **SQL-driven data warehouse**. Here, you use the SAP supplied, dedicated SAP HANA data warehousing tools, plus the generic SAP HANA tooling to build a data warehouse from scratch. The benefit to this approach versus the application driven approach is that customers can precisely define the architecture that suits them, starting small and growing into a full-scale enterprise-wide data warehouse. Compared to the application drive approach, the SQL driven approach encourages a more 'open' data warehouse where all data can be freely accessed by any tools outside the platform. Also, a freestyle SQL programming approach allows a developer to go beyond what the standard application tools would provide.

Of course, there are costs to these benefits, the most obvious ones are the requirements for skilled developers and also the time, effort and cost required to build a data warehouse from scratch.

Many customers have already begun this SQL driven data warehouse journey. Also, customers have realized that both approaches compliment each other where an application driven approach is mixed with an SQL driven approach to provide the optimal solution.

SAP BW is a ready-to-go, all-in-the-box data warehouse solution with built-in strong data governance, integrity, processes, controls, and so on. Going the SAP HANA route means that you have to build much of this from scratch. Many SAP BW customers rely on the strong built-in controls and data governance and do not want to develop this from scratch.

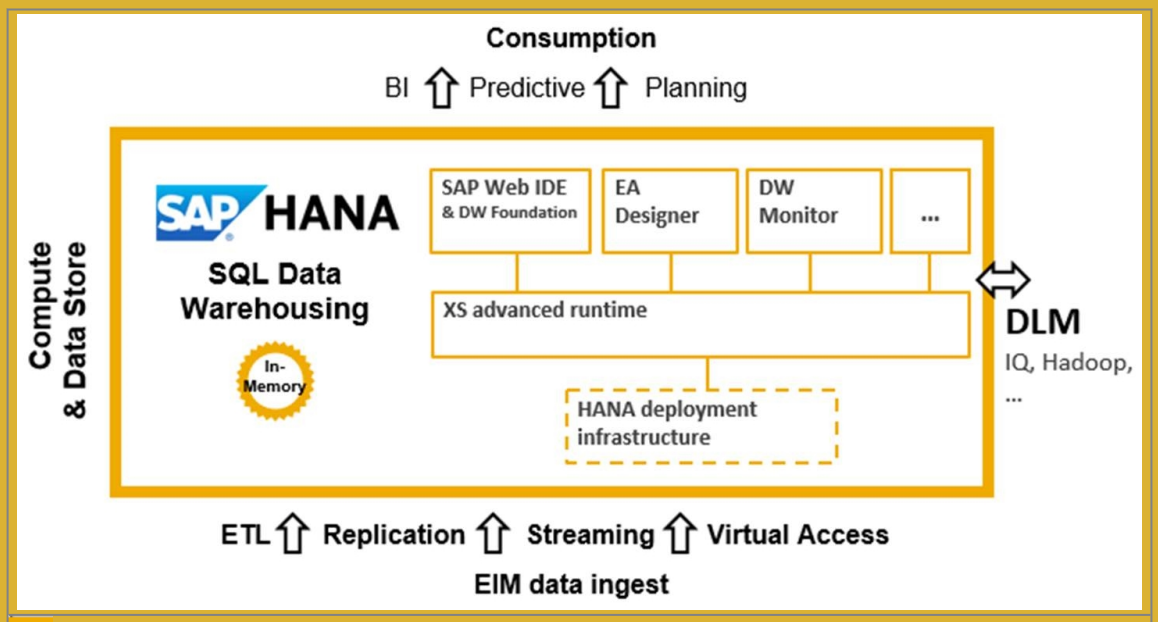


Figure 102: An SQL Driven Approach To Data Warehousing

The following list outlines the major data warehouse functions required and the SAP software component used for that function:

- **Compute and data store**: SAP HANA database is the primary data store and compute engine.
- **Data ingest**: Accessing and importing source data is done with the Smart Data Integration (SDI) for ETL services, Smart Data Access (SDA) for virtual data access and even Streaming Analytics for real time streamed data.
- **Design of the data warehouse**: SAP Enterprise Architecture Designer to plan and design your data warehouse.
- **Development of the data warehouse**: WebIDE for SAP HANA with the Data Warehousing Foundation add-on. The WebIDE is where developers spend most of their time, as activities like data definition, data transform, scheduling, and data preview are all done from there.

SAP HANA Data Warehousing Foundation

SAP HANA Data Warehousing Foundation (DWF) is a set of tools built by SAP that support the development and running of an SQL driven data warehouse on SAP HANA.

DWF is not shipped as a standard component of SAP HANA and must be installed separately.

Component of SAP HANA Data Warehousing Foundation

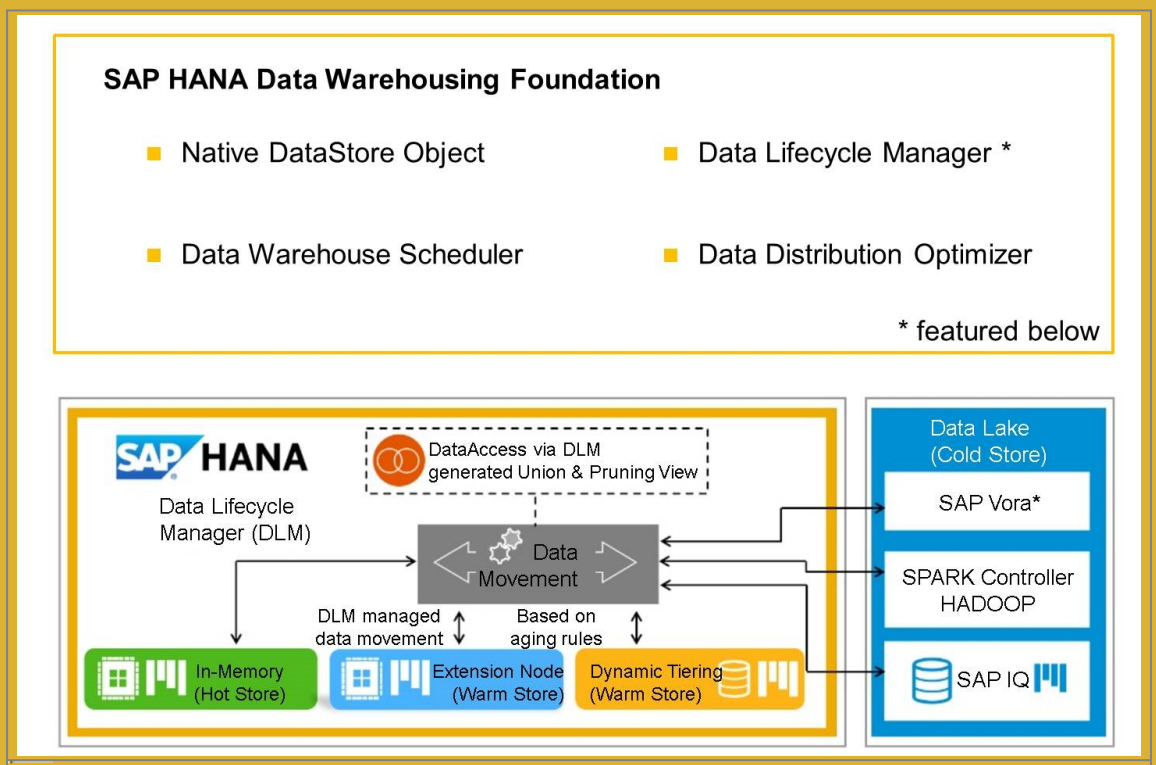


Figure 103: SAP HANA Data Warehousing Foundation

The following list describes the current components provided with SAP HANA Data Warehousing Foundation and their purpose:

- **Native DataStore Object (NDSO)** — The objective of the native datastore object (NDSO) is to provide a central data persistence object with advanced data load request handling. It can be directly compared with the standard SAP Business Warehouse (BW) advanced datastore object (ADSO). NDSOs can manage multi-source and multi-target delta loading, and data loads can easily be rolled back from anywhere in the NDSO flow. NDSOs are built and maintained using the Web IDE for SAP HANA.

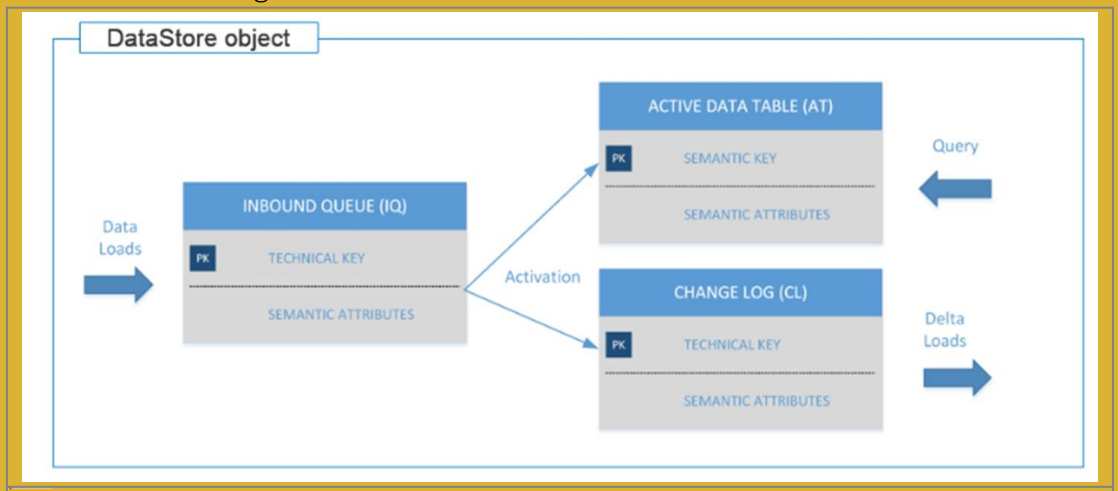


Figure 104: Native DataStore Object

- Data Lifecycle Management (DLM) — A tool to manage the relocation of 'aged'' data to other remote sources such as warm store (disk based storage) and cool store (data lakes), and vice versa. Enables SAP HANA administrators to model aging rules.
- The SAP HANA Data Warehouse Scheduler (DWS)— The SAP HANA Data Warehouse Scheduler (DWS) is a scheduling and process organizing tool built into Web IDE, to define task chains and design dependencies of individual data load processes. It uses flexible and simple scheduling instructions and includes, monitoring tools to observe job executions. Compare this directly to the Process Chain concept of SAP BW.
- SAP HANA Data Distribution Optimizer (DDO)— On a large scale, SAP HANA landscape data is spread out and stored across various nodes. This can mean that tables and partitions that are usually processed together, such as orders and returns, can find themselves located on separate nodes. The problem here is that joins must be carried out across nodes, which is not optimal for performance. Ideally, tables that are processed together should be on the same nodes. The SAP HANA DDO tool provides analysis of the use of join paths. This makes it easy to identify how tables have been processed together so they can be assigned to the same nodes for improved performance. With DDO, you create predefined relocation rules. These rules define the conditions under which data should be moved and to where the data should be relocated.

LESSON SUMMARY
You should now be able to:

- Describe SAP HANA Data Warehouse Foundation

Unit 5
Lesson 5

Building Native HANA Applications

229

LESSON OVERVIEW
In this lesson we will cover some of the basic concepts behind the creation of native SAP HANA applications using the XS engine.

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Describe the basics of native HANA applications

Introduction to Developing Native SAP HANA Applications

First, let's make sure we understand what is a **native** SAP HANA application.

When we refer to SAP HANA applications we need to differentiate between two types of applications:

- Applications that run on their own (non-HANA) application server and use SAP HANA only for the database services.
- Applications that are built and run directly on SAP HANA and use the SAP HANA built-in application server as well as the SAP HANA database services. These applications require no separate application server.

The following list outlines some applications that require their own application servers:

- SAP S/4HANA
- SAP Business Suite
- SAP BW/4HANA

As these applications are built using ABAP, they need to run on SAP NetWeaver Application Server. SAP HANA is not an ABAP application server and this is why the additional server is needed. In this scenario, SAP HANA provides the database services to the NetWeaver ABAP server. SAP HANA then becomes the bottom layer in a 3 tier stack made up of the GUI at the top then the application server layer and finally the database services layer at the bottom. In this case, the application are not native to SAP HANA. They are either **powered by** SAP HANA (if they are applications that can run on both SAP HANA and legacy databases), or they are **4/HANA**. (if they were built just to run on SAP HANA).

But an application that is built to run on the embedded application server of SAP HANA (XSA), is referred to as a **native** SAP HANA application. In the native application scenario we need just 2 tiers, the GUI and SAP HANA.

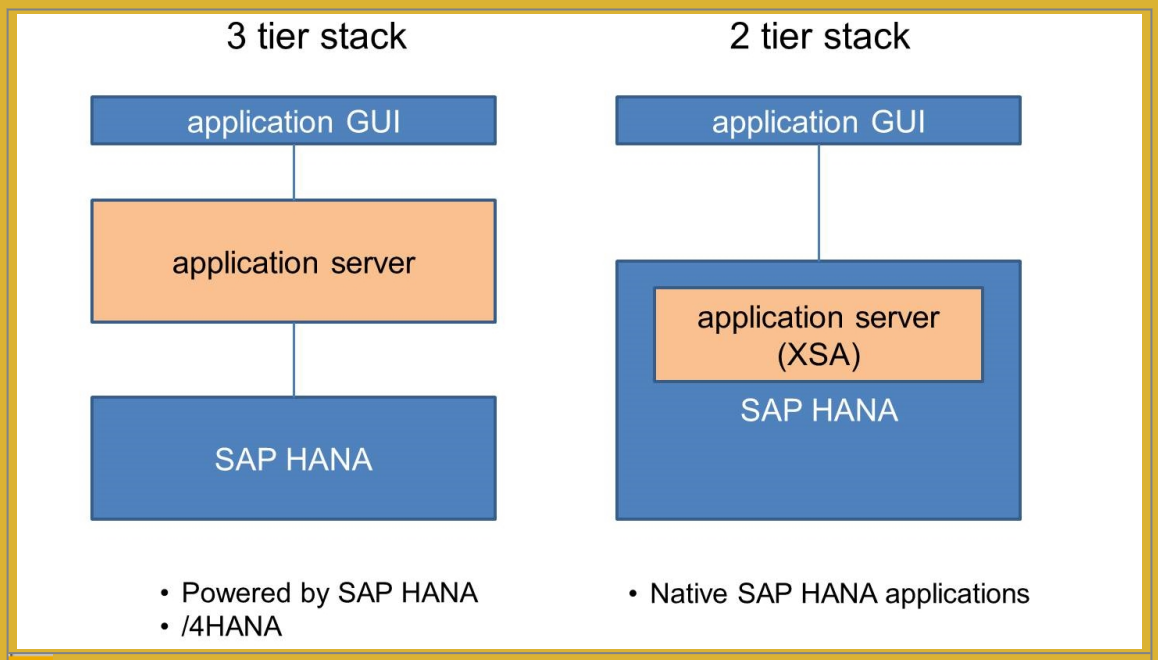


Figure 105: Native Versus Non–Native Hana Applications

Building Native Applications on SAP HANA

SAP HANA contains all the components required to develop and run applications in a simple two tier stack. There is no need to include an additional application server and this means less hardware and less complexity and of course, better performance without the connectivity between servers. SAP HANA supports modern programming languages and includes a full development and runtime environment for all types of applications. Any type of application can be built from simple dashboards to enterprise-wide applications.

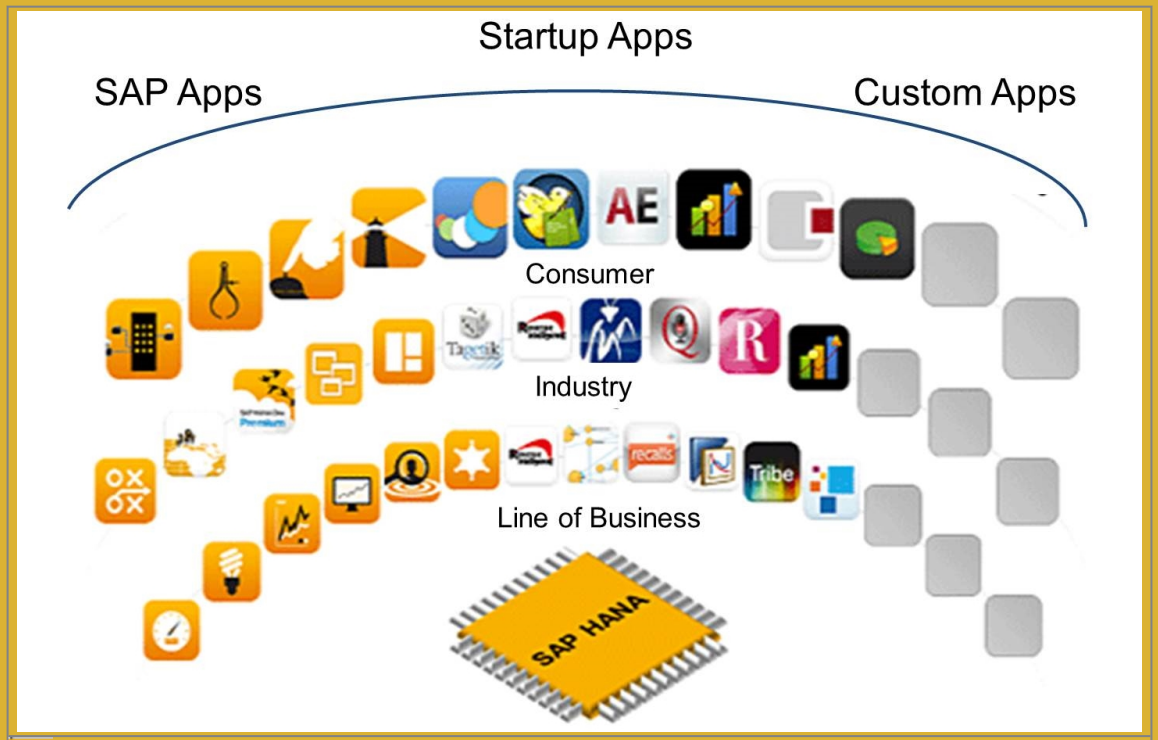


Figure 106: Native Applications on SAP HANA

With SAP HANA, you can build **scalable** applications that start small with minimal infrastructure requirements, few users, local usage and then later expand them to full-scale, enterprise–wide applications on large scale–out landscapes that support huge numbers of users on multiple devices.

SAP HANA application can be built to run on-premise or in the cloud, or a combination of both. They can also be built for on-premise deployment and then, without code changes needed, can be redeployed to the cloud. This also works the other way round — bring a cloud deployment on-premise.

SAP have built an active application development community who share ideas and code, enabling them to learn from each other. There are plenty of online resources to help you develop skills in native SAP HANA application development, and of course, there is the free of charge small footprint edition of SAP HANA called **SAP HANA Express edition**, so you can get started right away even using a basic specification laptop.

When you build applications in SAP HANA you have access to a rich application development platform which is tightly integrated with the in-memory SAP HANA database to enable high performance applications.

From XS to XSA

There is a little history to cover here just so we avoid confusion when you see references to XS and XSA.

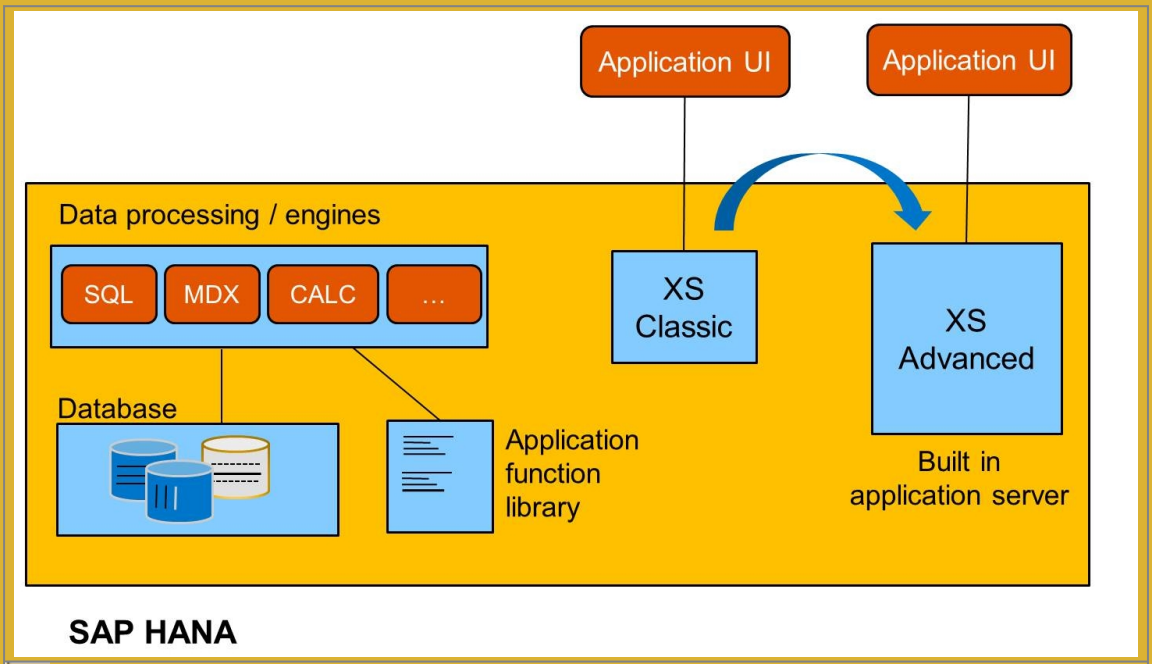


Figure 107: Evolution of the Built-In Application Server

Extended Application Services (XS) was first introduced with SAP HANA SPS05. Back then, it was positioned as a lightweight application server. It was never meant to be used to build heavy-weight, scalable enterprise applications. XS supports the development and run-time of Javascript and HTML-based applications. XS also includes a web server. It is fully integrated into SAP HANA and communicates directly with the SAP HANA database and, of course, processes data, and application logic completely in-memory for great performance.

Since SPS11, SAP delivered a significantly more powerful version of XS. This new version is called XS Advanced (XSA). Although XS and XSA have the same overall goal, they are

technically very different and can be considered two completely separate components. Applications developed with XS are not compatible with XSA and vice-versa.

Currently, XS and also XSA are installed side-by-side in SAP HANA 2.0 and both are currently supported. But **XSA completely replaces XS**, and in the future only XSA will be supported and in fact XS will be dropped from SAP HANA. Customers are encouraged to migrate their XS application to XSA as soon as possible using the supplied migration tools. Also, future developments should only use XSA. In fact, as of SAP HANA 2.0 SPS02, XS became *officially* deprecated.

Note:
XS is sometimes referred to as XS Classic. This is not an official SAP term but is popular online as a term.

So why did SAP implement a completely new application server framework (XSA) to replace XS?

De-couple applications from infrastructure — XSA is based on the common, open standard known as Cloud Foundry. This means that applications developed with SAP HANA XSA are cloud-ready and can easily be deployed either on-premise or to any cloud provider (SAP Cloud Platform, Microsoft Azure, AWS, Google Cloud Platform, and so on.) without changing the code. This gives greater choices to customers who can decide on their own deployment options and this safeguards their investment in the development effort and cost.

Support running multiple version of the same application — When an XS application is built it can only be deployed to one target runtime using fixed database schema coding. Using the new HDI approach we can now write schema-free code and this means applications can be easily redeployed multiple times to different containers with complete isolation. The approach supports running multiple versions of the same application in the same infrastructure.

Supports more languages — Instead of just one application language used in XS (JavaScript), XSA supports more development languages (JavaScript, Node.JS, JAVA. Plus you can also create your own custom run-time to support more languages such as C++, Python, Pearl and PHP.

Mix languages in one application — An important architectural change is that XSA supports a micro-services architecture. This is a modern approach to application development where applications can be built from multiple languages. The developer chooses the most effective development language for each part of the application and the run-times are combined to form a complete application. It also means it is easy to integrate other external services into your applications. And finally, it is now possible to configure each part of the application to consume more or less resources as needed. This is known as elastic computing.

Improve source code management — To enable the use of common industry standards for source code sharing and version control. XSA is fully integrated with the very popular Git/GitHub/Gerrit source code management solutions. SAP HANA no longer stores and manages the source code.

Improve security — With XS all users were assigned a database user with the required privileges to access database objects and data content. This approach, while nice and simple, is not acceptable to many customers who want separation between database access and application users.. With XSA, only the generated internal 'technical' database users are given access to the SAP HANA database. Application users' security is handled at the application layer which is turn passes the database requests to the technical user.

In a nut shell, XS was simply too restrictive, did not follow enough open standards, and could not support full-scale enterprise applications built for a multi-cloud / on-premise hybrid world.

Core Data Services

Before we can begin developing application code, we need the database objects in place, for example tables and views.

For native SAP HANA applications, SAP recommends that you build these using Core Data Services (CDS).

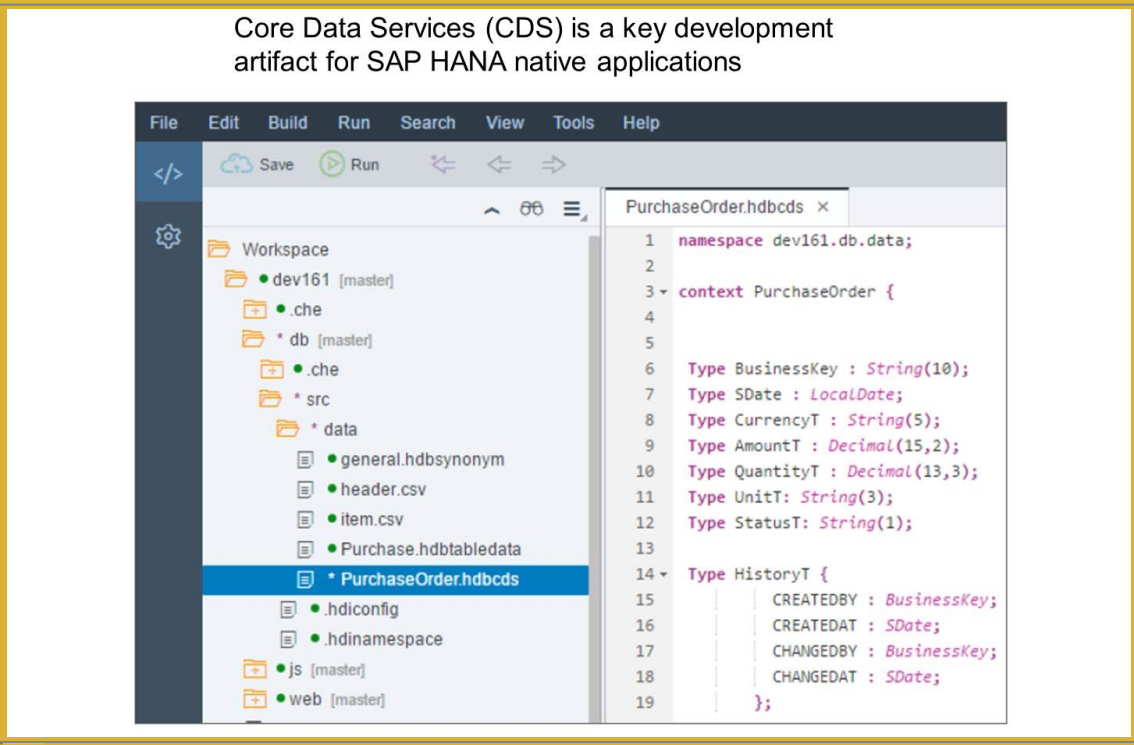


Figure 108: CDS - A Key Development Artifact

CDS uses a declarative language used to describe a data model used by an application and the three key objects that are built with CDS are as follows:

- Types — reusable definitions of single types or structured types (for example, amount or address – a structured type made up of street and city and country)
- Entities — tables defined from Types, but with the added ability to define *associations*, which are used to define the join conditions in a centralized way for easy maintenance.
- Views — similar to SQL views but with the added ability to add extra semantics to provide extra meaning to the columns that can be used by the consuming application.

The definition of types, entities, and views can be combined into a single file or split into individual files. All CDS files have the extension **`.hdbcds`**.

During the deployment, the underlying SQL database objects are generated automatically. It is not recommended to mix the development of SQL database objects with CDS development. Better to leave the generation of database objects to CDS.

However, then comes the frequently asked question: How do CDS and calculation views compare? Both are capable of producing views over tables.

The answer is that calculation views are optimal for OLAP scenarios (they use the OLAP engine) and offer many performance tuning features (pruning, and so on.) that are important when dealing with very high data volumes in aggregation scenarios. Also calculation views offer flexile consumption options for various external reporting clients such as ODBC, ODBO, and so on.

CDS views leverage the SAP HANA join engine and they offer a simple syntax for development, plus they offer associations for easy model building. Finally, CDS is the future for OData and Fiori development and are more suitable for coders who prefer to work in a text environment rather than with graphical tools (as with calculation views). SAP recommend using calculation views for pure BI cases. For hybrid applications (transactional mixing with analytical) CDS is a good approach.

CDS is built using the Web IDE for SAP HANA where you find both a text editor and a graphical editor. Most developers use the text editor but the graphical editor is useful if you are new to CDS and have not learned the syntax yet. You can easily switch between the two editors.

DDL Based Development

Although SAP provide the powerful Core Data Services (CDS) artifact for native SAP HANA application development of database objects, it is also possible to develop database objects using familiar artifacts found in most databases.

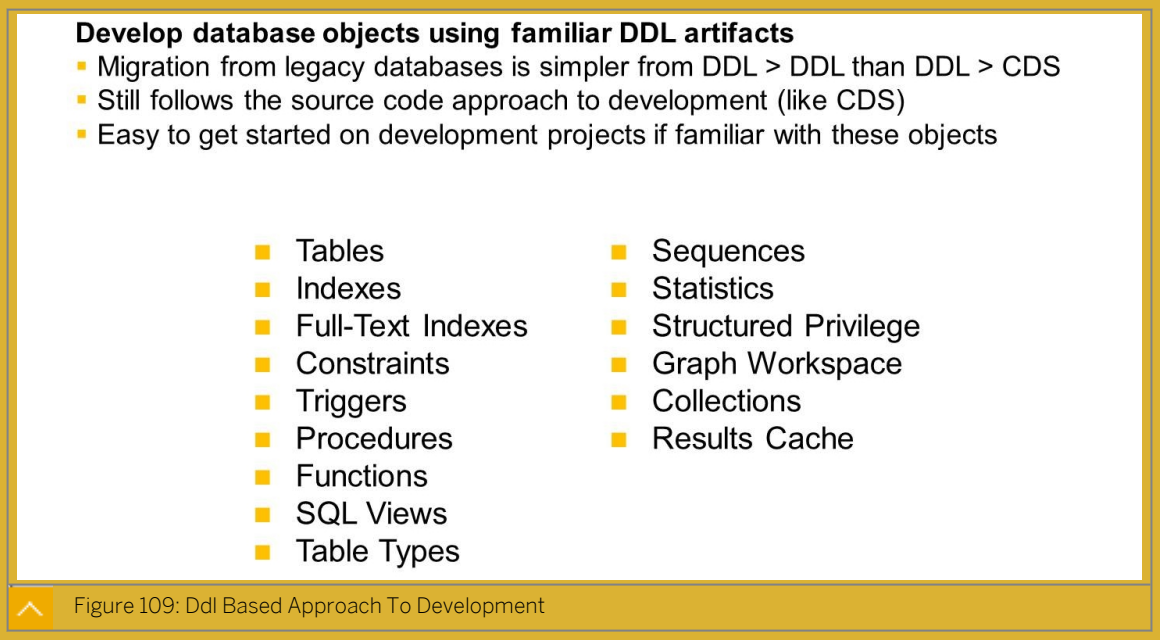


Figure 109: Ddl Based Approach To Development

Using these development artifacts means developers who are used to working in database development are more likely to be familiar with these and can get started quickly.

Just like with CDS, using these DDL artifacts still follows the recommendation to always develop using source code, as opposed to generating database objects directly from SQL statements.

DDL based artifacts are part of the HDI deployment approach so this means that a build includes these objects and all dependencies must be in place for successful build,

Each DDL artifact has its own extension. Here are some examples:

- Tables — .hdbtable
- Procedure — .hdbprocedure

- Constraint — .hdbconstraint

Sql Script For Database Development

To ensure the best possible performance of applications, developers should push as much processing to the SAP HANA database as possible. This can be achieved by writing as much code as possible in the database rather than in the application layer.

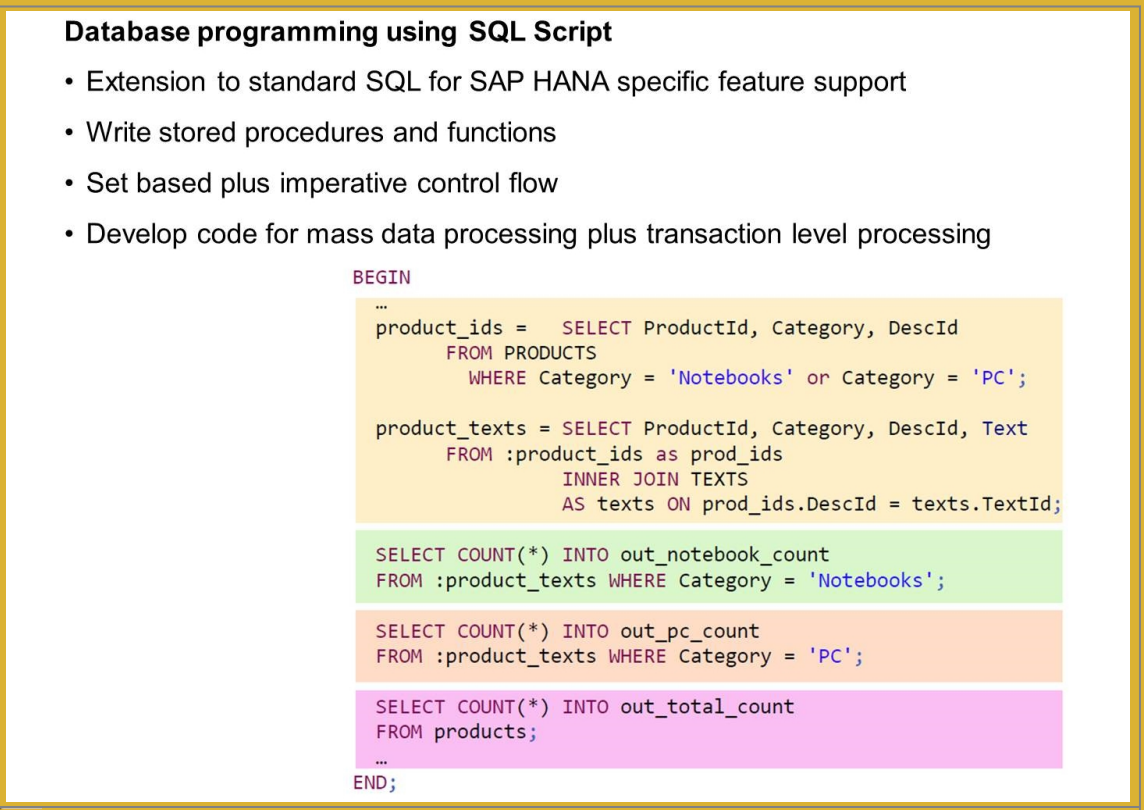


Figure 110: Sql Script For Database Development

SAP HANA provides the database language **SQL Script** to support this approach. Developers can write stored procedures and functions using a native SAP HANA language that builds on the well know ANSI-92 SQL standards syntax and adds SAP HANA specific features.

These additional features include variables to store internal results so that code can be simplified into smaller chunks but more importantly this encourages automatic parallelization. In fact, SAP HANA build SQL execution plans from SQL Script that automatically applies parallelization to all parts of the code where no dependencies exist. It is not required to request parallelization.

SQL Script is entered using the Web IDE for SAP HANA where you will find many productivity aids including:

- Code completion to speed up coding
- Real time error identification as you type (not just syntax errors but warnings to indicate where invalid objects are being referred to (tables columns)
- Source code debugger to step through code
- Code Analyzer to identify bad code (unused variables)
- Code libraries provided by SAP out of the box

The two main DDL artifacts where SQL Script is used are:

- Procedures (.hdbprocedure)
- Scalar and table functions (.hdbfunction)

Tools For Developers

Web IDE for SAP HANA

The Web IDE for SAP HANA is the main tool used by developers for XSA based development projects.

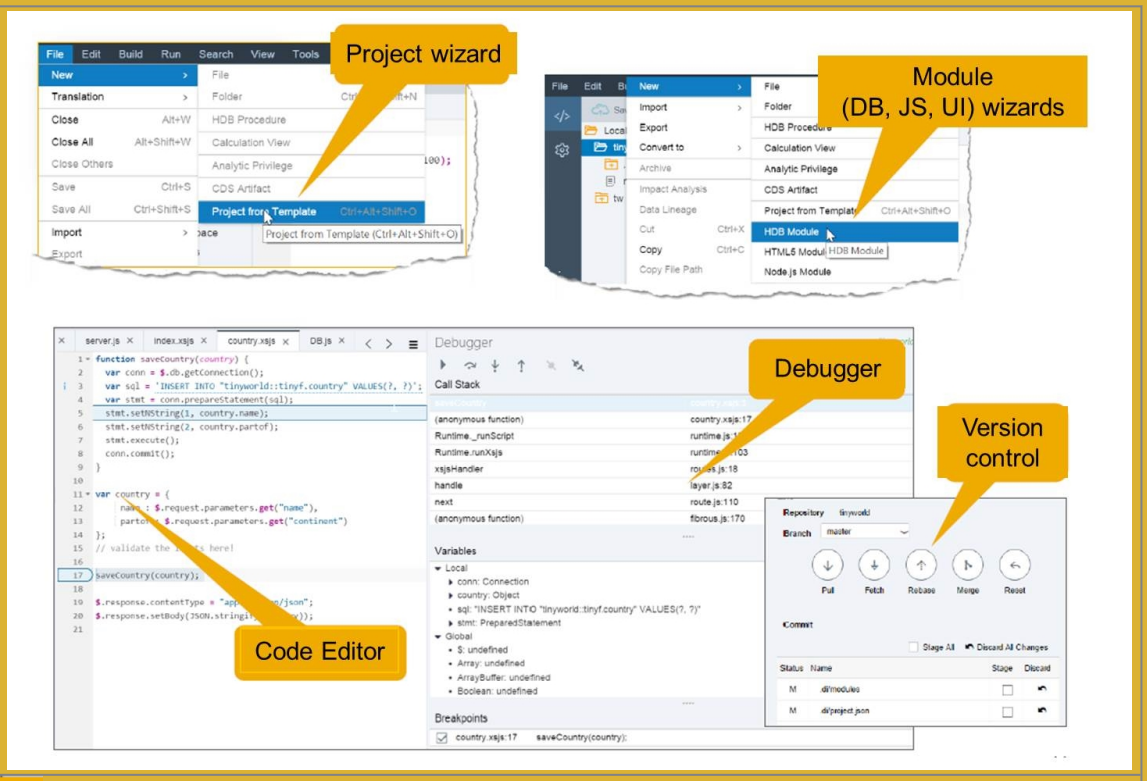


Figure 111: Web IDE for SAP HANA

The following features are provided and aimed at optimizing productivity:

- Project Wizard — get started with a new project using a built-in wizard that automates the creation of the key building blocks
- Code-specific modules — select from the provided modules to get started building code using the text and graphic editors relevant for each type of module
- Code Editor — a smart editor that provides real-time syntax checking, end of line code completion, code snippet templates, and database object dependency checks in real time
- Debugger — step through code and identify errors by viewing variable values
- Version Control — built-in Git control panel for managing source code versions and staging code

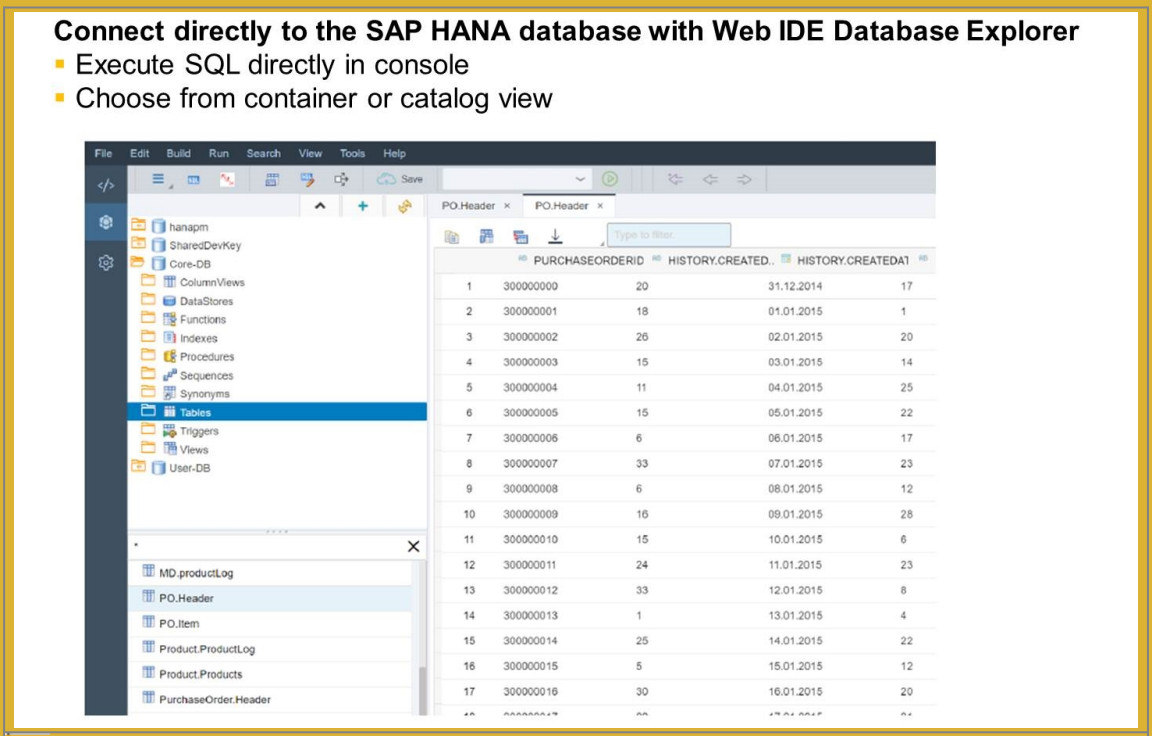


Figure 112: Connect Directly To Sap Hana Database

The Web IDE for SAP HANA includes a **Database Explorer** tool to directly connect to the database. This provides visibility of the database artifacts.

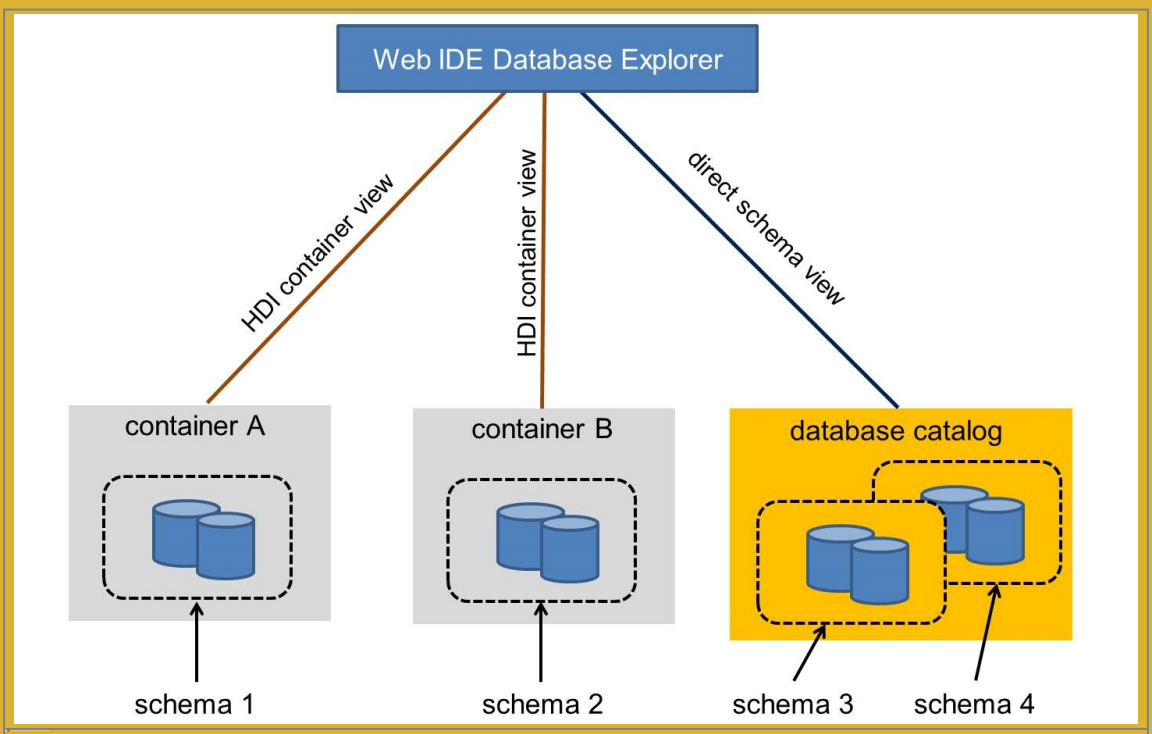


Figure 113: The Hdi Container or Schema View

It is possible to connect directly to the database using the classic catalog/schema approach, or by connecting to the database via the containers. Using the catalog/schema approach provides an unrestricted view of the entire database (subject to having the correct SQL

authorizations). The container approach is useful to view the local database objects that are built as part of the application.

Source Code Management

If you work in application development you already appreciate the challenges of working in a team where each of the members is working on a different part of the application. Team members lock each other when they need to access common objects, or even worse their developments are overwritten by others.

Things are made more complex when the application needs to provide new features very frequently and at the same time bugs and fixes must be dealt with urgently on the current versions of the application.

Finally, a key challenge is tracing the history of not just one piece of code but all the dependant objects at any moment in history.

With the first generation application server (XS), source code was managed completely inside SAP HANA. A source code repository was used and developers checked in and checked out their code.

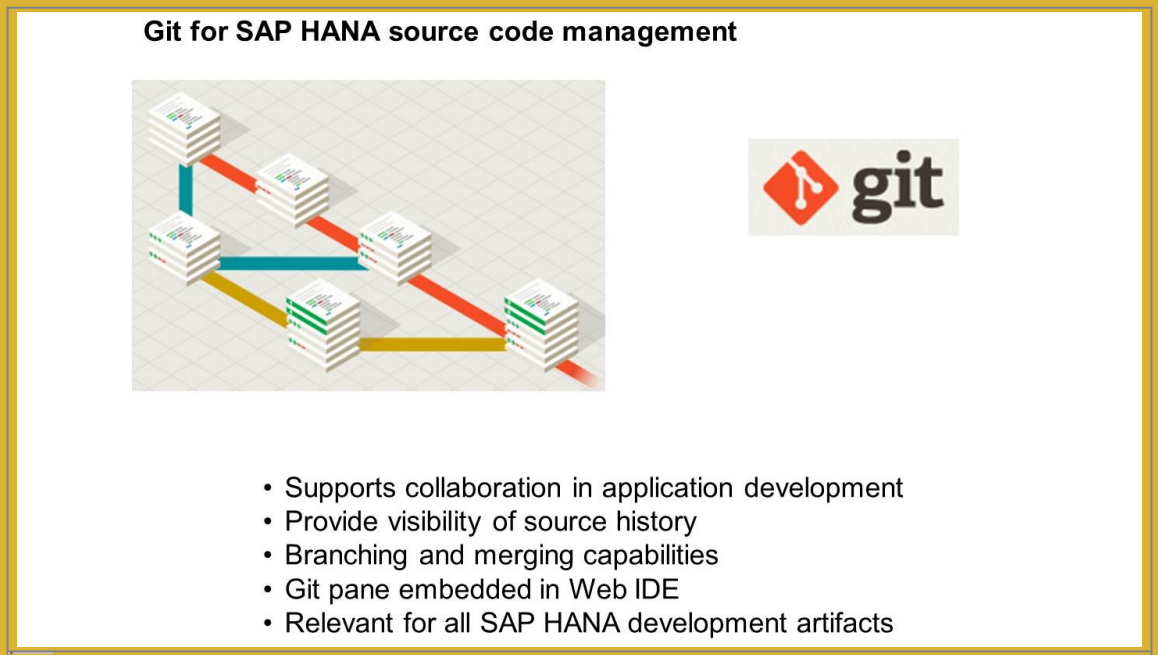


Figure 114: Git For Source Code Management

SAP HANA XSA uses Git for source code management. This includes all application and database code plus calculation views (as technically they are also stored as text files). Git has established itself as the leading source code management tool and is open source. Many application developers are already using Git and so they are able to get started right away in a familiar environment.

Git provides sophisticated branching and merging capabilities that **does not rely on locking** objects. Multiple developers can work on their own branches of the code in parallel and then merge the results when they are finished. Developers can pickup the version of the application at any stage of development.

There are code review tools that allow conflicts to be managed, for example when a source file is modified by two developers and we need to figure out which version to use, or perhaps how we might manually adopt both the code from one into the other.

SAP have developed a Git console right inside the Web IDE for SAP HANA, so that all source artifacts can be seen with a Git status, such as staged, committed etc It is recall old versions of source code as the history paths for all code branches is shown graphically.

In addition to Git, you can also implement Gerrit or GitHub which provides additional capabilities especially in the area of community collaboration which even extends outside your organization. A lot of publicly available code snippets are available on GitHub and Gerrit for downloading. Plus of course, it is easy to push your code to GitHub / Gerrit for sharing.

So, with a well established industry standard for source code management, SAP decided to give up providing these tool inside SAP HANA and use Git. SAP can now focus on adding value in other areas.

Note:
There are many online resources to learn Git, including some great videos. SAP provide some of these.

Building Applications on XS Advanced

XSA's architecture is based on the following:

- Cloud Foundry
- Multiple Development Languages
- Microservices Architecture
- Git
- HANA Deployment Infrastructure (HDI)

Cloud Foundry

Cloud Foundry is an open-source cloud platform-as-a-service (PaaS), on which developers can build, deploy, run, and scale applications on public and private cloud models. One of the key requirements for XSA was the desire to unify the architecture of solutions built in the cloud and on-premise. Cloud Foundry provides scalability options and flexible runtimes that are needed in the cloud environment. SAP HANA Cloud Platform is based on Cloud Foundry. For on-premise HANA we utilize only the basic aspects of Cloud Foundry. However, what is key is that we can now unify the developments of cloud and on-premise applications. This eliminates the wide gap that was present between on-premise and cloud using XS. One of the main advantages of Cloud Foundry is that we can build applications using multiple development languages and also microservices architecture (see next). This means that we are not restricted to JavaScript, but can also work with other languages including Java, node.JS, and C++. The emphasis on Cloud Foundry is portability of applications.

Multiple Development Languages

SAP HANA XSA provides full native development support for the application languages **JavaScript** (on Node.js), and **Java** (on TomEE).

Of course, XSA provides runtime engines to support these languages but also many other popular runtimes are supported including PHP, Pearl, Ruby, Python, C++.

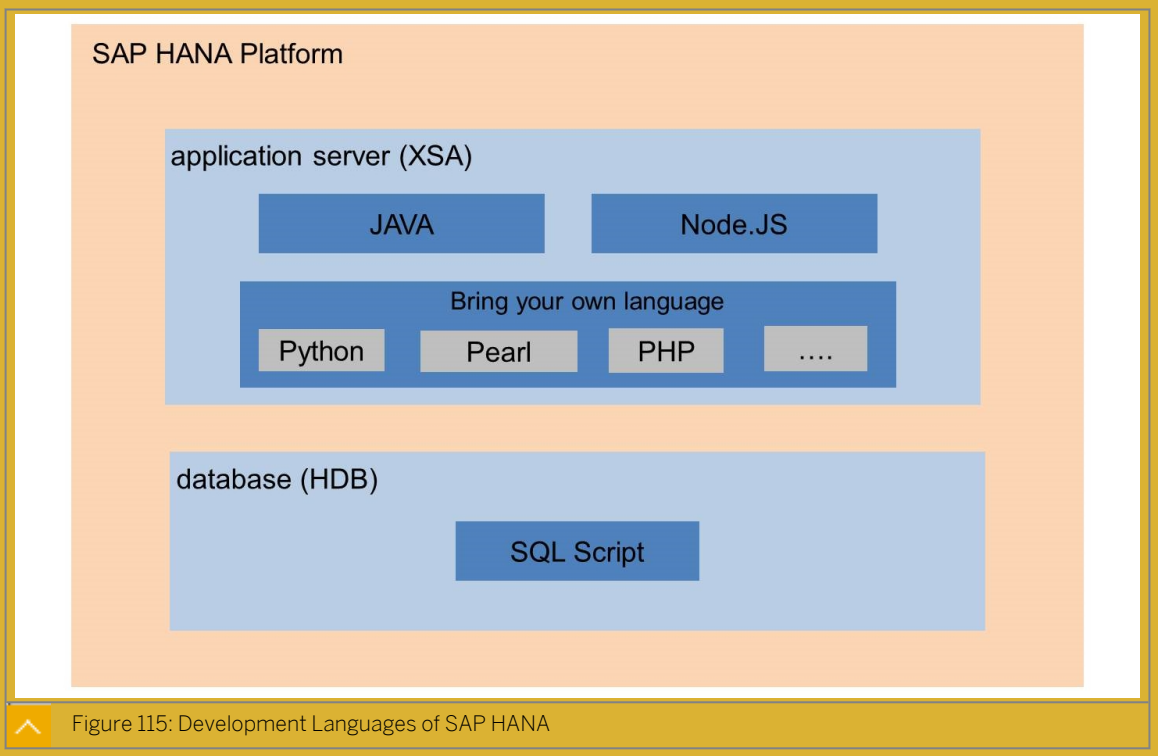


Figure 115: Development Languages of SAP HANA

To support **OData**, which is the SAP-recommended protocol for REST-based data access,. XS advanced applications model the database objects that must be exposed to client requests by annotating CDS views or defining OData artifacts; the applications can then make use of these generic data providers that interpret these models.

SAP HANA now supports **Hibernate**. Hibernate is an open source object relational mapping (ORM) tool that provides a framework to map object-oriented domain models (such as sales orders or vendors) to relational databases. SAP HANA is one such relational database that is supported. Many developers now work in Hibernate in their Java projects and are familiar with this modern tool. Hibernate provides an abstraction layer between the application and the database to make it possible to address, access, and manipulate objects without having to consider how they are related to their data sources.

Microservices Architecture

Another key innovation in XSA is the use of microservices architecture. Microservices architecture is a modern approach to building applications from individual modules that are combined using REST APIs. The modules can be developed in any language (see the figure, Development Languages of SAP HANA) and are reusable across applications. Each module can be regarded as an individual service.

Microservices architecture allows us to follow the industry trend for building applications using the best tools and languages for the job (this is referred to as 'bring your own language').

Microservices enable better scaling and memory management, which means that each runtime runs in its own isolated container and can be individually tuned. It also means that if an individual service fails, it does not bring down the entire application. When an XSA application is called, the **XSA Router** orchestrates the different runtime modules that make up the complete application. Microservices have their own development and maintenance lifecycle, so you can upgrade one service independently of other services.

Git

Git is one of the industry-leading source code version control frameworks. It is already used by a large number of developers to manage their software lifecycle. It can be used privately for internal projects or publicly for open-source projects where developers can contribute and share enhancements in a safe and controlled manner.

SAP have chosen Git as the new source code and design time object repository for XSA-based development. SAP Web IDE for SAP HANA has capabilities that allow the developer to interact directly with Git, in order to commit, stage, fetch, merge development artifacts. We no longer store source code and other design time artifacts in the HANA repository, as we did with XS.

HDI

With XSA comes a brand new approach to deployment. We call this HANA Deployment Infrastructure (HDI).

At design time, we write code and also develop database objects (such as tables, functions) in source code files that are organized into **modules** using Web IDE for SAP HANA.. A module looks like a top level folder and specifies the language of the source code such as Node.JS, or HTML. There is always at least one module dedicated to the database artifacts. This is the HANA Database Module (HDB).

A group of modules sit under a development **project**. To generate the run-time objects from our design time files, we **build** an application from all the source code sitting in the modules under a project. A build ensures that all object dependencies are checked and if there was even one small error in any source file, or a missing file, the entire application build fails. This is called the 'all-or-nothing' approach and ensures that all source code files of an application are committed together. It is possible to build isolated run-time objects by selecting an individual source file. But again, its dependencies are strictly checked and any required, but not yet available objects, cause the build to fail.

The generated run-time database objects of an application are organized in a **database container**. These containers provide isolation of the database objects that are used in different versions of the same application. We can redeploy the same application, with different features and enhancements and each applications version has its own database container. This means that we can continue to run existing applications without disruption, while offering newer versions, or variations of the application that might not suit all consumers. We can deploy versions of the same applications to cloud or on-premise. This is a big change from how we deployed with XS. With XS we could only deploy one version of an application at a time..

Source Code Organization

Within each module, we can freely create additional sub folders to organize our source code.

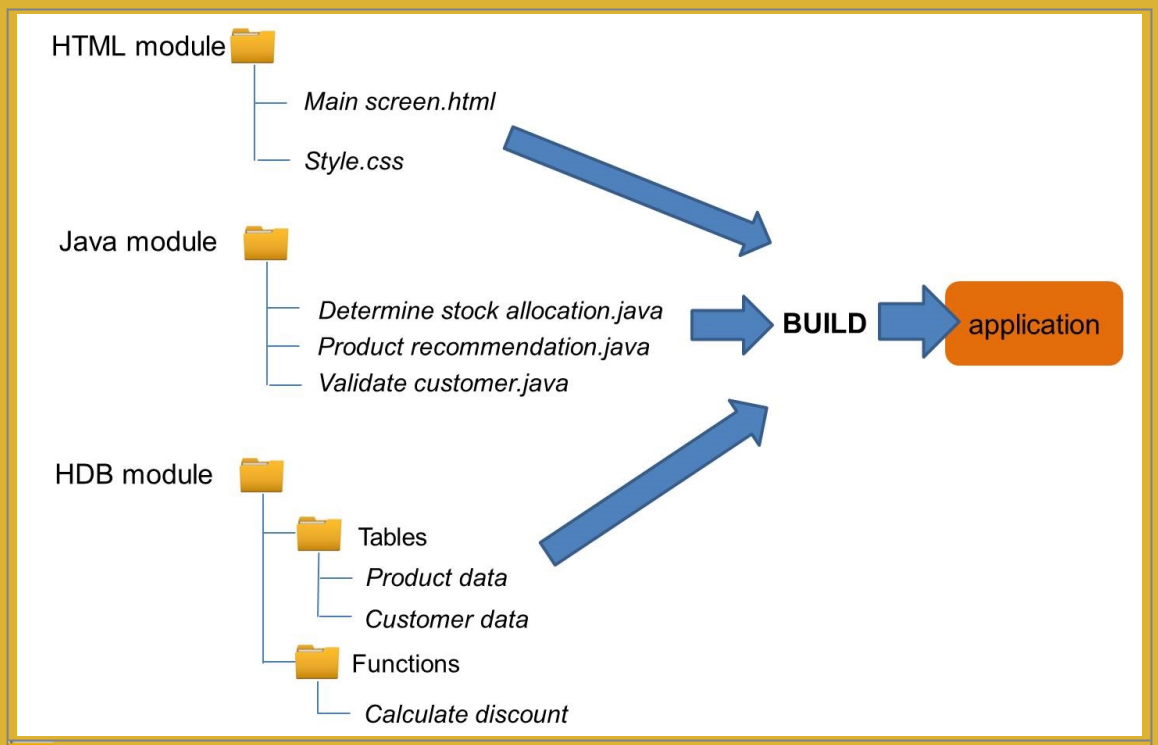


Figure 116: Source Code Organization In XSA

For example, under an HTML5 module we might create a folder called *STYLES* and in there we store the .css files. Also under the HTML5 module we might have a folder called *IMAGES* where we store the graphics used on the interface (such as buttons).

When using Git, you also see the Git status control icons alongside each source file to show whether it has been staged or committed.

It is easy to **export** any source file, or a complete folder, or even a whole project, so it can easily be **imported** to another folder in the same SAP HANA system or even an completely different SAP HANA systems. This makes the sending of source files to support teams for debugging, very easy. The source files are text based files so they are very portable and require no special software to open them and display contained code.

Note:
To learn more about native application development on SAP HANA, you may want to consider the course HA450, *Application Development for SAP HANA*.

XS to XSA Application Migration

Since SAP HANA 1.0 SPS11 there are two very different application development and run time architectures available: XS Classic (XS) and XS Advanced (XSA). With SAP HANA 2.0, SAP strongly recommends to only use XSA.

However, customers have already built many applications in XS, the original application development and run time environment introduced in the very early releases of SAP HANA. What should they do with these XS applications? Do they still run on SAP HANA 2.0?

Customers have three choices as to how they proceed:

- Continue to run the application under XS. SAP HANA 2.0 supports existing XS-based applications with no adjustments needed.

- Redevelop the application in XSA using newer tooling and multiple languages.
- Migrate the entire XS application to XSA.

The key motivation for the migration of XS based applications to XSA is to realign them to the more powerful and flexible architecture. Also, XSA is the only architecture that will be supported going forward.

XSA Advanced Migration Assistant

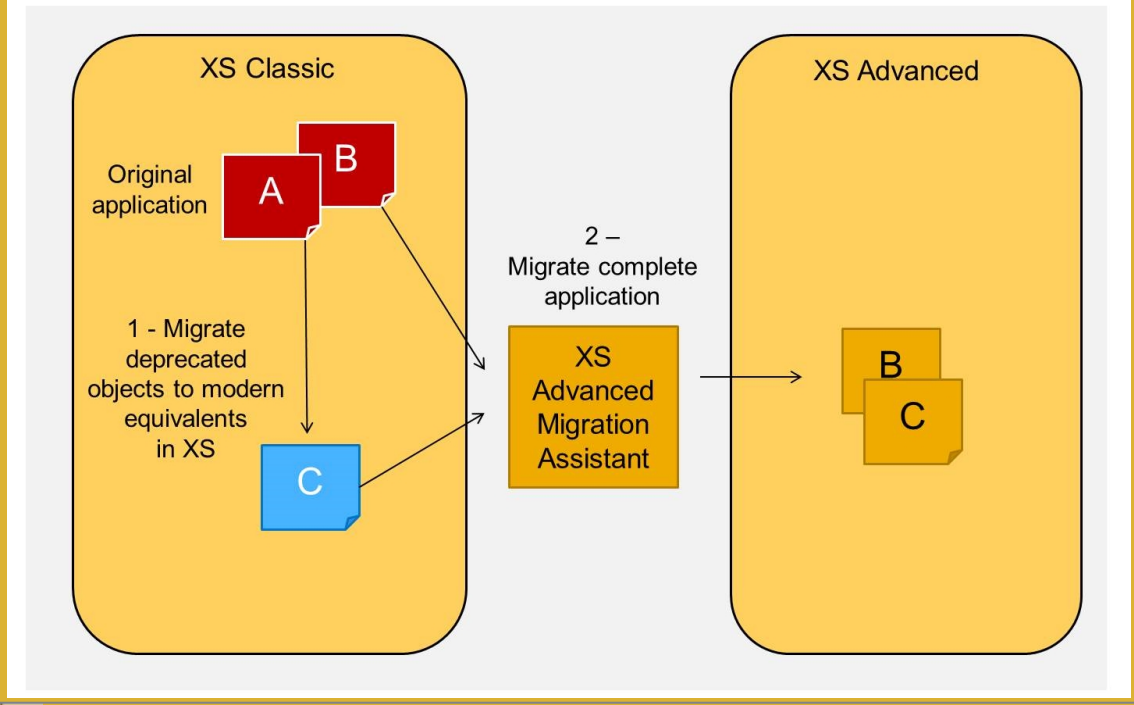


Figure 117: Migrate XS Application to XSA

To support the migration process, SAP provide tooling. The main tool is called **XS Advanced Migration Assistant** . This tool automates the migration of all objects that belong to an application so they become XSA objects.

However, the XS Advanced Migration Assistant is not able to migrate deprecated objects such as:

- Scripted Calculation Views
- Decision Tables
- Attribute Views
- Analytic Views
- XML based Analytic Privileges
- Application Function Library (AFL) models

You must first migrate those objects using the tooling that is found in the SAP HANA Studio. The migration of those objects converts them to modern objects that are recognized in XSA and are supported going forward. For example, Scripted Calculation Views are converted to Table Functions. XSA does not recognize the deprecated objects.

Migration Process

Once the deprecated objects are migrated, you can then begin the migration of complete XS applications.

Migration is a very detailed process with many steps. The steps are summarized as follows:

1. Install XS Advanced Migration Assistant.
2. Configure the connection to the XS classic and XS advanced systems.
3. Migrate the XS classic application.
4. Review the detailed report generated by the migration assistant.
5. Upload and deploy the migrated application to XS advanced.

To support the migration process, SAP deliver a real-life demonstration of the XS Advanced Migration Assistant using the SAP HANA Interactive Education (SHINE) demo application. We show you how to use the XS Advanced Migration Assistant to migrate one of the demo applications from XS classic to XS advanced. SAP have developed this SHINE capability in order to illustrate and explain the typical challenges you are likely to encounter when migrating XS classic applications.

Application Function Libraries

You can dramatically increase performance by executing complex computations in the database instead of at the application sever level. SAP HANA provides several techniques to move application logic into the database, and one of the most important is the use of application functions delivered by SAP.

Application functions are like database procedures written in C++ and can be called with parameters to perform data intensive and complex operations. SAP delivers ready-made functions that are grouped into an **Application Function Library (AFL)**.

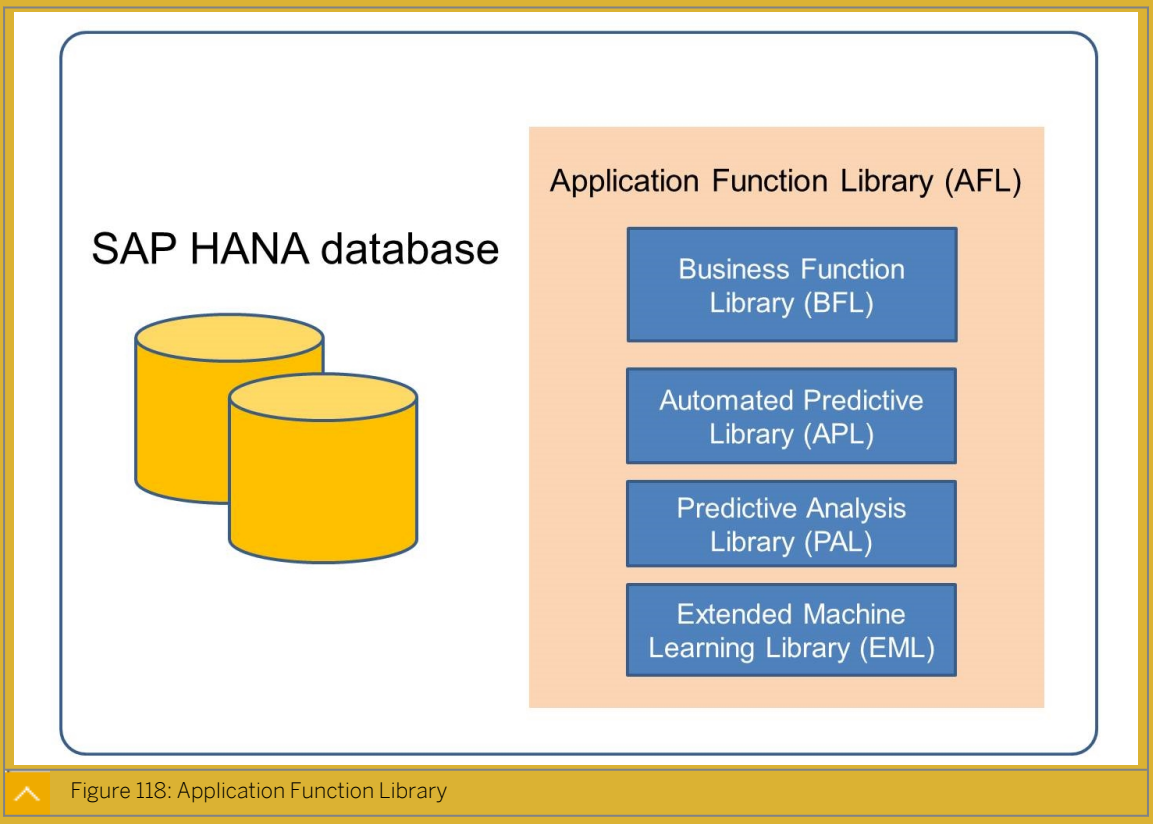


Figure 118: Application Function Library

Application Function Library (AFL) uses the following libraries:

- Business Function Library (BFL) — contains pre-built parameter-driven functions that can be embedded in applications to perform common business functions.
- Predictive Analysis Library (PAL) — contains optimized predictive algorithms that are supplied with SPA HANA and run natively inside SAP HANA in-memory database.
- Automated Predictive Library (APL)

  provides access to the additional algorithms that are packaged with SAP Predictive Analytics tool, but again, the algorithms run natively in SAP HANA.
- Extended Machine Learning Library (EML) — introduced with SAP HANA 2.0 SPS02. functions, provides access to external machine learning libraries. The first external library is Google TensorFlow, but more may be added.

,

SHINE - The SAP HANA Reference Application

SAP HANA Interactive Education (SHINE) is a fully built model application developed and maintained by SAP and is delivered free of charge and can be installed in SAP HANA. SHINE enables new SAP HANA developers to learn and develop applications running on top of the SAP HANA platform.

SAP HANA SHINE

- Pre-built, fully working reference application based on XSA
- Learn best practices from SAP
- Learn how to work with key data processing techniques (text, spatial etc.)
- Constantly updated by SAP to include new features of SAP HANA

Figure 119: SAP HANA SHINE

SHINE content is designed and built upon the EPM (Enterprise Procurement Model) framework developed by SAP and it includes all of the data models, tables, views, dashboards and so on, with a real enterprise use case. It also comes with embedded documentation that describes the building blocks of the applicator from a technical perspective as well as operating instructions. There are also plenty of online resources to guide you with the installation and exploration of SHINE.

SHINE include analytics, text, spatial, and even transactional (CRUD) applications. The idea is that you learn from these fully functioning applications rather than struggle yourself to get something working.

SHINE even comes with its own data generator so you can generate and load data into the tables used by SHINE.

Note:
There is a version of SHINE for XS as well as a version for XSA so be sure to download the correct version.

Developers should download SAP HANA Express edition and install SHINE so that they have a quick start environment to build their native SAP HANA application building skills.

247

Task 1: Launch an XS Advanced Native Application

Exercise Objectives

In this exercise you learn how to launch an application that was built completely on SAP HANA that runs on XS Advanced, and also how to locate the source objects in the development view of Web IDE.

Business Example

You would like to familiarize yourself with an application that was built completely on SAP HANA XSA.

> Note:
> In this exercise, when values include ##, replace these characters with your own student number.

1. Launch the SHINE XS Advanced application using the shortcut *SHINE for XS Advanced.url* in the HA100 folder, and log on using the details provided in the table, *Logon Details*.

   Table 14: Logon Details

Field	Value
User	**`STUDENT##`**
Password	**`Training1`**

2. Launch the **Sales Dashboard** and review the analysis content and the existing sales orders.
3. Create a new sales order choosing any business partner and any products and quantities.

Task 2: Review the Source Objects the XSA Application

1. Import the SHINE XSA project into your workspace.
   The project archive *hana-shine-xsa.zip* is located in your *HA100* folder.
2. Locate the calculation views used by the SHINE application.
3. Locate the data files used to provide sample data to the SHINE application.

> Note:
> Do not attempt to build the application, it will fail because we did not define all the project settings and parameters. The purpose of this task is to describe where the source objects are located and how they are organized.

Task 1: Launch an XS Advanced Native Application

249

Exercise Objectives

In this exercise you learn how to launch an application that was built completely on SAP HANA that runs on XS Advanced, and also how to locate the source objects in the development view of Web IDE.

Business Example

You would like to familiarize yourself with an application that was built completely on SAP HANA XSA.

> Note:
> In this exercise, when values include ##, replace these characters with your own student number.

1. Launch the SHINE XS Advanced application using the shortcut *SHINE for XS Advanced.url* in the HA100 folder, and log on using the details provided in the table, *Logon Details*.

   Table 14: Logon Details

Field	Value
User	**`STUDENT##`**
Password	**`Training1`**

   a) Navigate to the Windows folder: *Favorites* → *HA100* → *URLs*.

   b) Double-click the file *SHINE for XS Advanced.url*.

   c) Log on using the details from the table, *Logon Details*.

   d) Close the *What's New* dialog pop-up box using *OK*.

2. Launch the **Sales Dashboard** and review the analysis content and the existing sales orders.

   a) Choose *Sales Dashboard* tile.

   b) Close the pop-up dialog box by selecting *Continue*.

   c) Review the analysis content on the *Overview* tab.

   d) Select the *Details* tab to view the existing sales orders.

3. Create a new sales order choosing any business partner and any products and quantities.

   a) Choose *New*.

b) From the *Select a Business Partner* drop down, select any business partner.

c) From the *Select a Product* drop down, select any product.

d) To save the order, choose *Create*.

e) Check that your order now appears in the detail list.

Task 2: Review the Source Objects the XSA Application

1. Import the SHINE XSA project into your workspace.
   The project archive *hana-shine-xsa.zip* is located in your *HA100* folder.
   a) In Web IDE Development View, right-click the *Workspace* folder, and choose *Import → File or Project*.
   b) Choose *Browse* and navigate to the Windows folder *Favorites → HA100*.
   c) Highlight the file *hana-shine-xsa.zip* and choose *Open*.
   d) Choose *OK*.
2. Locate the calculation views used by the SHINE application.
   a) Expand the folders *hana-shine-xsa > core-db > src > models* and you see the calculation views.
3. Locate the data files used to provide sample data to the SHINE application.
   a) Expand the folders *hana-shine-xsa > core-db > src > data > loads* and you see the raw data files used to load the tables (.csv), alongside the source files that provide the loading instruction (.hdbtabledata).

Note:
Do not attempt to build the application, it will fail because we did not define all the project settings and parameters. The purpose of this task is to describe where the source objects are located and how they are organized.

LESSON SUMMARY

You should now be able to:

- Describe the basics of native HANA applications

Learning Assessment

253

1. What are features of SAP Business Suite powered by SAP HANA?

   Choose the correct answers.

   - [X] A Includes Embedded Analytics
   - [X] B ABAP is replaced with JAVA
   - [X] C Combine operational analytics and transaction processing
   - [X] D Performance improvement

2. What are features of SAP S/4HANA?

   Choose the correct answers.

   - [X] A Optimized for SAP HANA but maintains compatibility with other databases
   - [X] B Embedded Analytics
   - [X] C Rewritten ABAP code optimized for SAP HANA
   - [X] D SAP HANA Live included to provide real time operational analytics

3. What are key capabilities of SAP HANA Analytics Cloud (SAC)?

   Choose the correct answers.

   - [X] A Landscape monitoring
   - [X] B Predictive
   - [X] C Business Intelligence
   - [X] D Planning

4. Which of the following does SAP HANA use to connect to reporting tools?

   Choose the correct answers.

   - [X] A IDOC
   - [X] B ODBC
   - [X] C JDBC
   - [X] D ODBO

5. In which key areas can we expect improvements when running SAP BW powered by SAP HANA?

   Choose the correct answers.

   - [X] A Tighter security
   - [X] B Data loading performance
   - [X] C Superior data quality
   - [X] D Reporting performance

6. Why do I install SAP HANA Data Warehousing Foundation add-on?

   Choose the correct answer.

   - [X] A To enhance the performance of SAP BW powered by SAP HANA
   - [X] B To provide the essential components needed to develop an SQL-driven custom data warehouse
   - [X] C To move my SAP BW/4HANA on-premise solution to the cloud

7. Which interface client is used for XSA application development?

   Choose the correct answer.

   - [X] A SAP HANA Web-based Workbench
   - [X] B SAP Web IDE for SAP HANA
   - [X] C SAP HANA Studio
   - [X] D SAP Web IDE

8. Applications developed with XS classic must be migrated to XSA before they can run in SAP HANA 2.0.

   Determine whether this statement is true or false.

   - [X] True
   - [X] False

9. What does XSA based application development use for source code management?

   Choose the correct answer.

   - [X] **A** HDI
   - [X] **B** Node.JS
   - [X] **C** Git
   - [X] **D** Cloud Foundry

Learning Assessment - Answers

256

1. What are features of SAP Business Suite powered by SAP HANA?

   Choose the correct answers.

   - [X] A Includes Embedded Analytics
   - [X] B ABAP is replaced with JAVA
   - [X] C Combine operational analytics and transaction processing
   - [X] D Performance improvement

   Correct!

2. What are features of SAP S/4HANA?

   Choose the correct answers.

   - [X] A Optimized for SAP HANA but maintains compatibility with other databases
   - [X] B Embedded Analytics
   - [X] C Rewritten ABAP code optimized for SAP HANA
   - [X] D SAP HANA Live included to provide real time operational analytics

   Correct!

3. What are key capabilities of SAP HANA Analytics Cloud (SAC)?

   Choose the correct answers.

   - [X] A Landscape monitoring
   - [X] B Predictive
   - [X] C Business Intelligence
   - [X] D Planning

   Correct!

4. Which of the following does SAP HANA use to connect to reporting tools?

   Choose the correct answers.

   - [X] A IDOC
   - [X] B ODBC
   - [X] C JDBC
   - [X] D ODBO

   Correct!.

5. In which key areas can we expect improvements when running SAP BW powered by SAP HANA?

   Choose the correct answers.

   - [X] A Tighter security
   - [X] B Data loading performance
   - [X] C Superior data quality
   - [X] D Reporting performance

   Correct!.

6. Why do I install SAP HANA Data Warehousing Foundation add-on?

   Choose the correct answer.

   - [X] A To enhance the performance of SAP BW powered by SAP HANA
   - [X] B To provide the essential components needed to develop an SQL-driven custom data warehouse
   - [X] C To move my SAP BW/4HANA on-premise solution to the cloud

   Correct!

7. Which interface client is used for XSA application development?

   Choose the correct answer.

   - [X] A SAP HANA Web-based Workbench
   - [X] B SAP Web IDE for SAP HANA
   - [X] C SAP HANA Studio
   - [X] D SAP Web IDE

   Correct!

8. Applications developed with XS classic must be migrated to XSA before they can run in SAP HANA 2.0.

   Determine whether this statement is true or false.

   - [X] True
   - [X] False

   Correct!

9. What does XSA based application development use for source code management?

   Choose the correct answer.

   - [X] A HDI
   - [X] B Node.JS
   - [X] C Git
   - [X] D Cloud Foundry

   Correct!

UNIT 6 Appendix: Continuing your SAP HANA Learning Journey

Lesson 1

UNIT OBJECTIVES

- Develop your SAP HANA knowledge

Developing your SAP HANA Knowledge

261

LESSON OVERVIEW
In this lesson we will introduce the various follow-on courses and other educational resources available to help you continue your SAP HANA learning journey.

LESSON OBJECTIVES
After completing this lesson, you will be able to:

- Develop your SAP HANA knowledge

Further SAP HANA Courses

Although this is an appendix, please try to make time to cover this in class, even briefly. Students are usually very grateful to hear a knowledgeable instructor provide trusted advice on further education options. We need to ensure that students have a clear idea of how they can move forward with the development of their HANA knowledge. SAP HANA is broad and deep, so we need to ensure that students don't fumble around looking for courses and other education resources (or even worse spend money on the wrong courses or courses with weak non-SAP content). But to prove you haven't suddenly switched into a potentially annoying end of class, selling mode, you should mention the free education resources such as the YouTube HANA Academy Channel and also OpenSAP. You may even like to remind students of SAP HANA Express (the free downloadable version of SAP HANA for your laptop).

SAP HANA education content is organized into five tracks, which support the roles of typical SAP HANA project team members.

These tracks are as follows:

- SAP HANA Modeling
- SAP HANA Administration and Operations
- SAP HANA Application Development
- SAP HANA Data Provisioning
- SAP Business Warehouse powered by SAP HANA / SAP BW/4HANA

The key courses for each track are listed in the following sections and are intended to provide you with suggestions for your next SAP HANA course.

SAP HANA Modeling

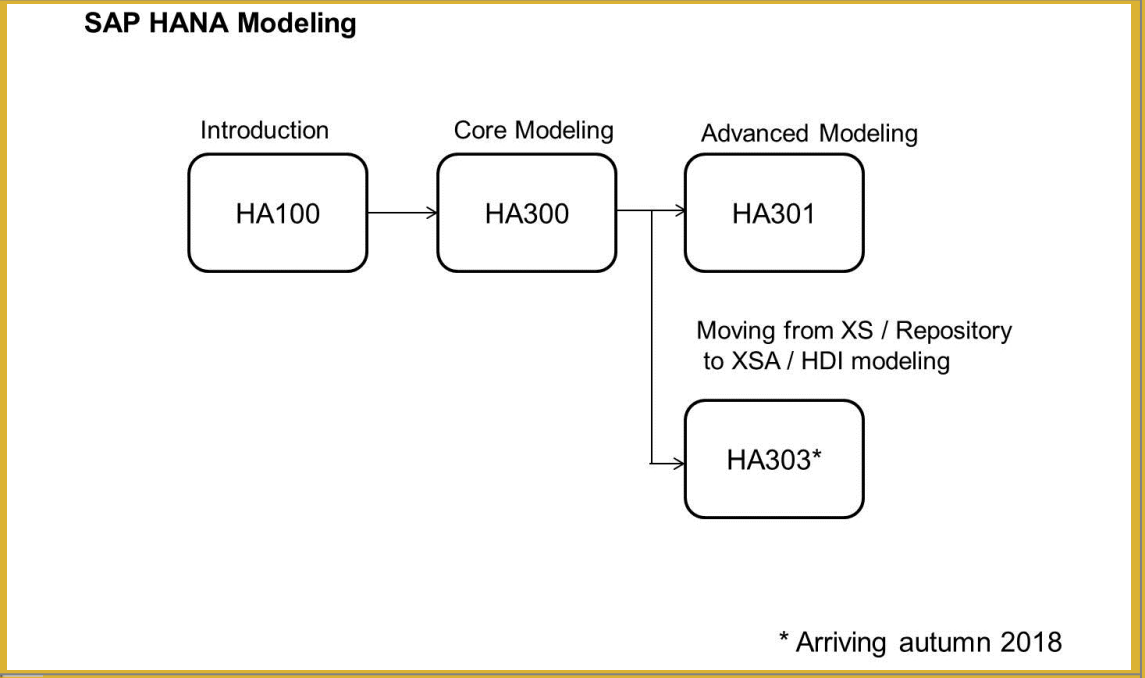


Figure 120: SAP HANA Modeling — Key Courses

SAP HANA Administration and Operations

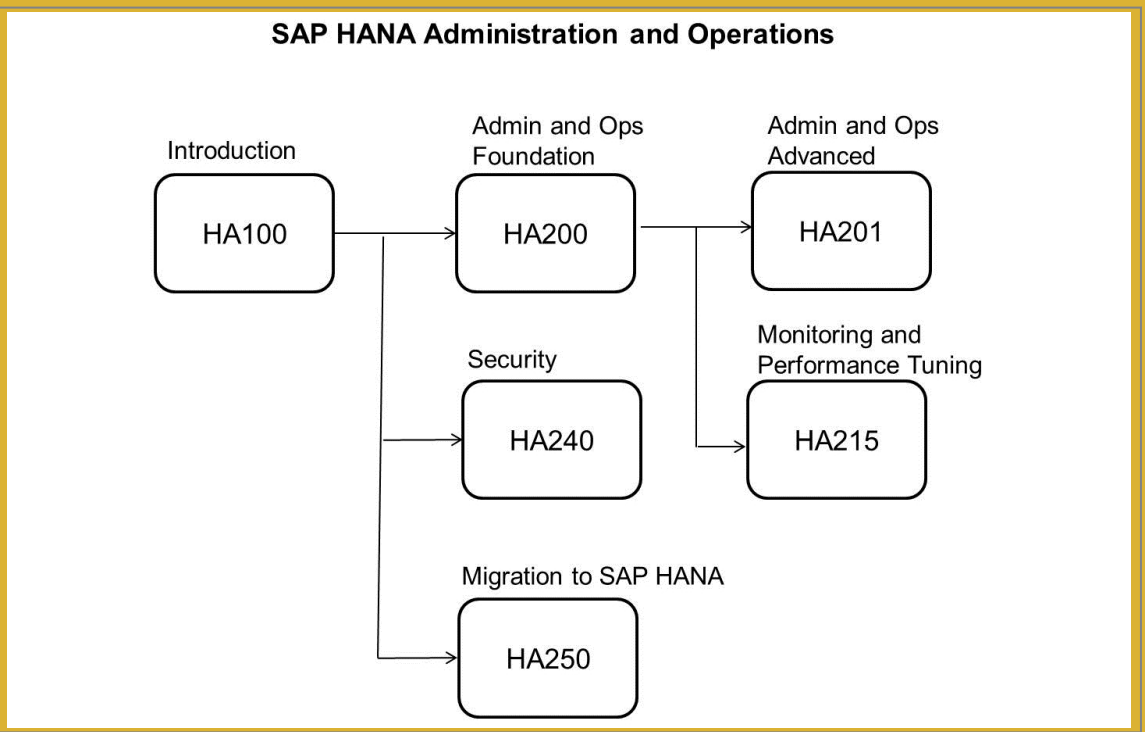


Figure 121: SAP HANA Administration and Operations — Key Courses

SAP HANA Development

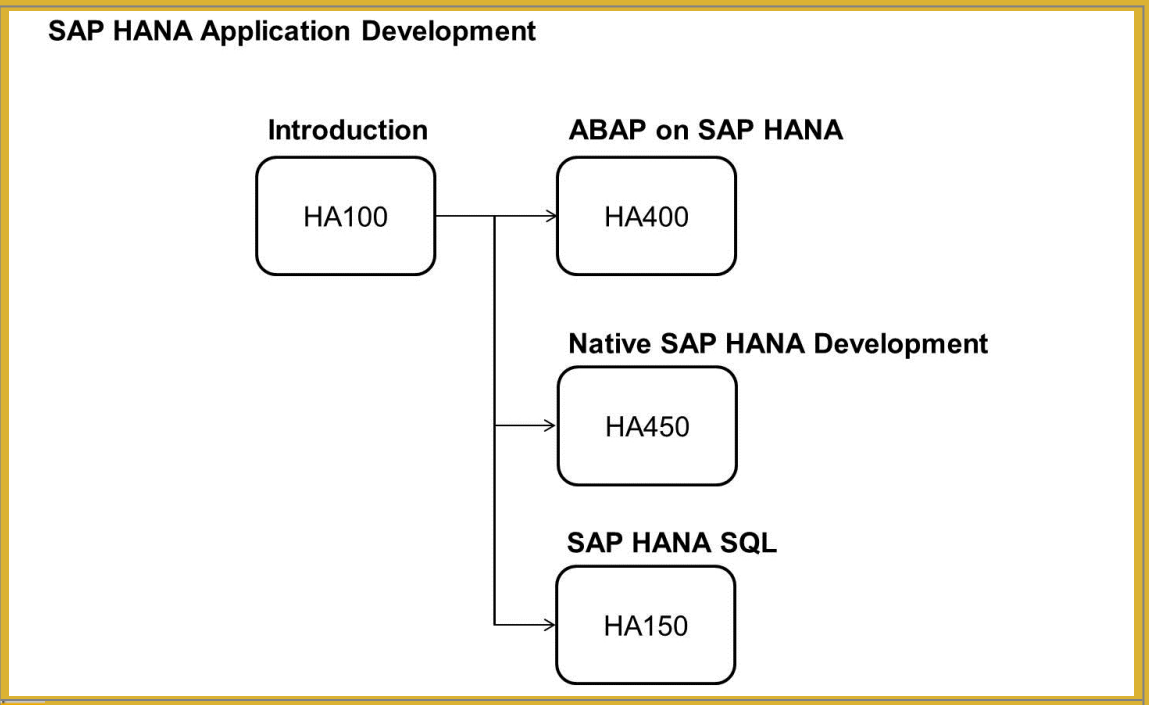


Figure 122: SAP HANA Application Development — Key Courses

SAP HANA Data Provisioning

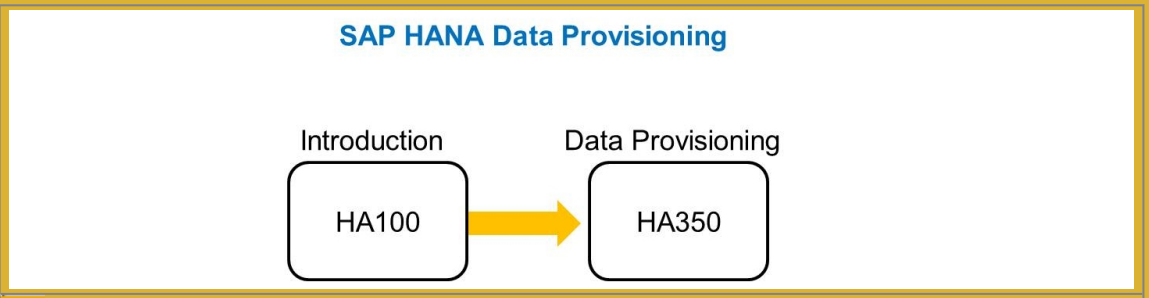


Figure 123: SAP HANA Data Provisioning — Key Courses

SAP BW powered by SAP HANA and SAP BW/4HANA

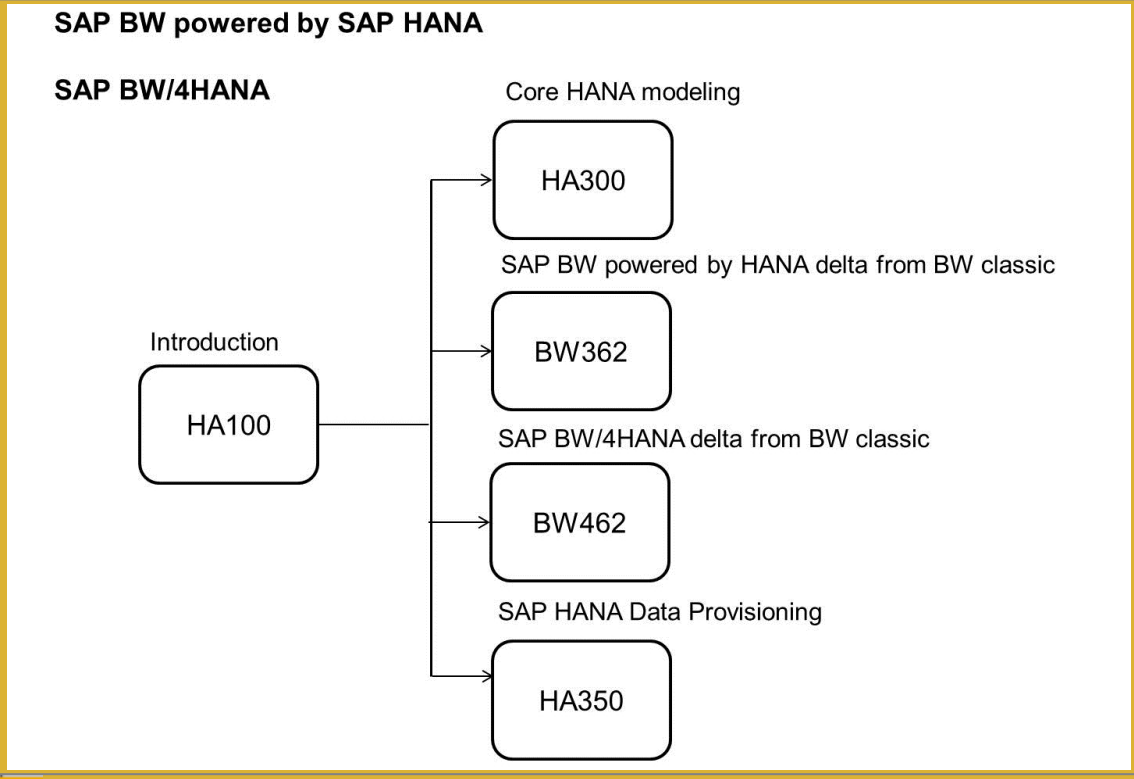


Figure 124: SAP BW powered by SAP HANA — Key Courses

Learning Journeys

For each track, SAP provide an **SAP Learning Journey**, which helps you identify which piece of education content you should follow and in the sequence that is most sensible. These include suggestions to take not only the courses described previously, but also other types of helpful education content such as OpenSAP workshops, Learning Hub content and Learning Rooms, and certifications to create a complete end-to-end journey from novice to expert.

The SAP Learning Journeys are found at: http://help.sap.com/learningjourneys.

Consumption Options for SAP HANA Courses

There are three modes for consumption of SAP HANA courses. They are as follows:

- **Classroom**

  These are public scheduled events that run at an SAP Training Center. Alternatively, in most regions, you can have the class delivered privately on your own site, or a serviced office (usually Regus) near you.

- **SAP Live Class (also known as Virtual Live Class (VLC))**

  These are public scheduled events where all students are connected via the internet, with full two-way face to face communication with the instructor. You are sent the full course materials in advance as well as the SAP HANA training system access details. You are part of a live event using web cams with the same duration as a classroom event covering exactly the same content.

- **E-learning**

E-learning presents exactly the same materials as the other consumption options, but you take these self-study courses in your own time. You access the e-learning version of all SAP courses by subscribing to **SAP Learning Hub**. You then have access to all slides, demos (which have audio commentary), assessments,, plus the full course handbooks.

Note:
When you see references to ILT, this means Instructor Lead Training, which means the class is lead by an instructor in real-time as opposed to self study. Classroom and VLC events are both ILT. E-learning is not ILT.

Please refer to the SAP Education shop at training.sap.com for the very latest information on all courses, locations, prices, and schedule.

Tailored Courses

In most regions it is possible to have a class tailored to your exact requirements and run just for you, either on your site, at an SAP Training Centre, or via SAP Live Class. This is called a Customer Specific Training (CST) event. We will work with your team to identify the ideal topics to cover and this can also include topics across multiple courses to produce a course that focuses on only topics that are relevant to you.

Please be aware — there is a minimum number of participants required for CST events.

Speak to your local SAP Education team to find out more.

SAP Learning Hub

SAP Learning Hub is SAP's**cloud based learning platform** and provides 24/7 access to SAP education content in multiple formats and in multiple languages. The content is enhanced by social learning and peer collaboration. SAP Learning Hub is a subscription based service and there are various editions ranging from the full enterprise edition to individual topic area editions.

SAP Learning Hub consists of three key components:

- Learning Content — SAP courses (the same as used in the classroom)
- Learning Rooms — SAP expert-led communities to enable collaboration with SAP experts (including SAP instructors) and other fellow learners
- SAP Training Systems — Fully configured training servers where you can work through all course exercises just like in the classroom

Let's describe each component in a little more detail.

Learning Content

Learning content includes access to SAP courses, the same as we use in the classroom. You consume the same slides and demos with expert recorded narration and also the course handbooks are available to you, plus on-line assessments to help you gauge your progress. You can work on as many courses as you wish. You can dive into any topics in any course at any time as you need to meet your on-demand learning requirements.

Refer to the course maps provided earlier for a list of SAP HANA related courses that you will find in SAP Learning Hub content.

SAP Learning Rooms

SAP Learning Rooms are SAP instructor-led collaborative communities that support you in your learning journey. In the Learning Rooms, you can do the following:

- Ask questions anytime to SAP instructors and also review the archive of previous questions that were already answered
- Watch additional videos developed and posted by the instructors and other SAP experts
- Attend regular live sessions hosted by experts covering key topics, and ask your questions, live
- Replay the Live Session recordings
- Read blogs from experts
- Check your HANA knowledge using our exam preparation aids and sample questions

SAP Learning Rooms are ideal for students who are preparing for certification. There are assessments and tips available to ensure that you are ready for the exams.

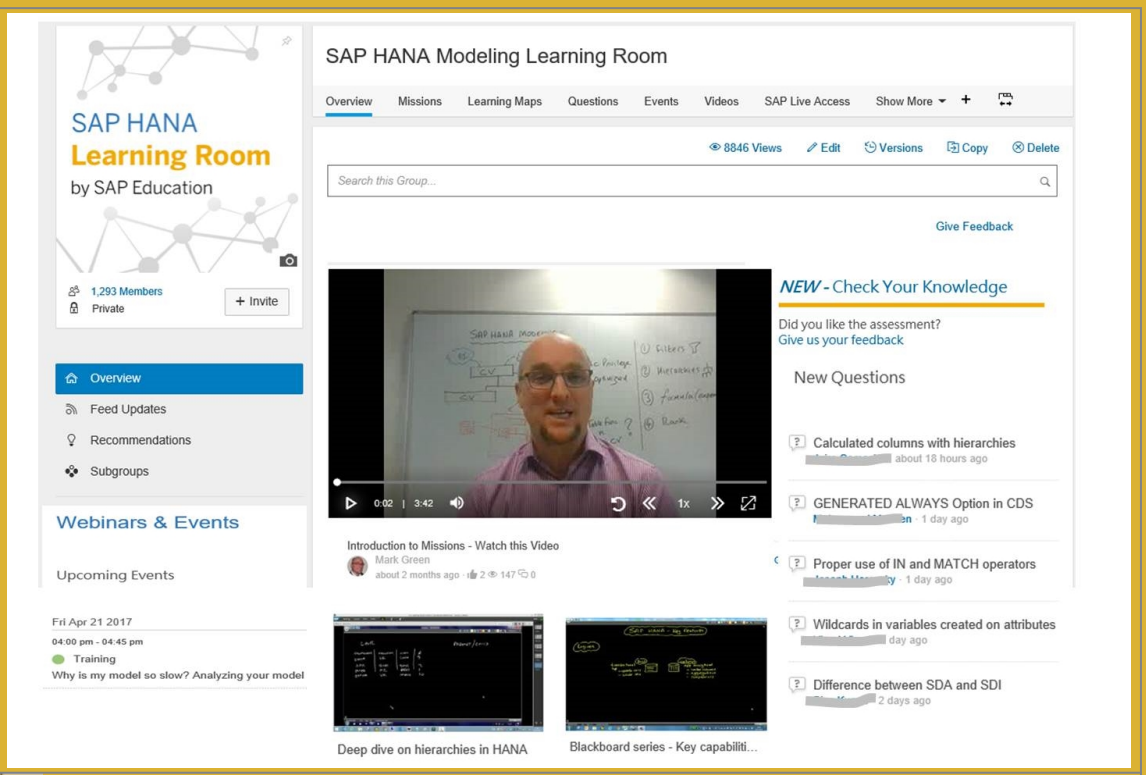


Figure 125: HANA Learning Rooms

There are many different SAP Learning Rooms and they align to popular SAP topics, such as SAP HANA modeling and SAP HANA administration. SAP Learning Rooms are available as part of a subscription to SAP Learning Hub at no additional cost. Learning Hub subscribers can join as many Learning Rooms as they wish. There are currently almost 100 Learning Rooms available and more are being added all the time.

SAP HANA Learning Rooms available in the SAP Learning Hub

Learning Room Title	Description	Learning Objectives
SAP HANA Implementation & Modeling	This room will provide you with a collaborative environment to help you work towards taking the SAP Certified Application Associate - SAP HANA certification exam. Of course, if you don't plan to take the certification, but still wish to learn more about modeling with SAP HANA then this room is for you	• Develop a broad view of SAP HANA, it's position as a platform, the architecture, and development tools • Develop detailed knowledge in the area of SAP HANA Implementation, Modelling, and Data Provisioning. • Prepare yourself for SAP HANA certification: **C_HANAIMP**XXX - SAP Certified Application Associate - SAP HANA
ABAP for SAP HANA	This room will provide you with a collaborative environment to help you work towards taking the ABAP for HANA specialist certification exam. Of course, if you don't plan to take the certification, but still wish to learn more about ABAP for HANA, then this room is for you. With expert instructors on hand to answer any questions you have, appropriate self-study material, extra content on the most important topic areas, and live online sessions where you will have the chance to interact with an instructor and each other, this room is a great way to work towards certification.	• The ABAP for HANA Certification room (**E_HANAAW**XXX - SAP Certified Development Specialist (Edition 2015) - ABAP for SAP HANA) is intended to guide learners through the topic areas needed for the ABAP for HANA Specialist certification exam.
SAP HANA Administration and Operations	This room will provide you with a collaborative environment to help you work towards taking the SAP Certified Technology Associate - SAP HANA certification exam. Of course, if you don't plan to take this certification, but still want to learn more about SAP HANA Technology, then this Learning Room is for you.	• Prepare for the Certification exam: **C_HANATEC**XXX - SAP Certified Technology Associate - SAP HANA
SAP BW powered by SAP HANA	This room will provide you with a collaborative environment to help you work towards taking the SAP Certified Application Specialist - SAP BW powered by SAP HANA exam. Of course, if you don't plan to take this certification, but still want to learn more about SAP BW on SAP HANA, then this Learning Room is for you.	• Gain a fundamental understanding of the possibilities and the positioning of SAP BW powered by SAP HANA, with an ultimate objective to apply this knowledge as a solution consultant in a team setting. • Prepare for the Certification exam: **E_HANABW**XXX -

Figure 126: SAP HANA Recommended Learning Rooms

SAP HANA Training System Access

A subscription to SAP Learning Hub provides exclusive rights to access the training systems that we use for all SAP HANA classes. To get started you need to purchase an access voucher. Each system is a fully-configured SAP HANA landscape and is not shared by others. It is a complete SAP HANA landscape just for your use. SAP Live Access is ideal when you want to work through all the exercises in a class, or even just to explore or perhaps prepare for certification.

SAP HANA – Live Training System Access

Course Code	Course Title	Target Audience
HA100	SAP HANA – Introduction	Application Consultants Project team members
HA150	SQL Script: Basics and Advanced for SAP HANA	Application Consultants, Development Consultants
HA200	SAP HANA – Installation & Operations	Technology Consultants System Administrators
HA215	SAP HANA – Monitoring and Performance Tuning	Technology Consultants System Administrators
HA240	SAP HANA Authorizations, Scenarios & Security Requirements	Technology Consultants System Administrators
HA250	SAP HANA – Migration to SAP HANA	Technology Consultants System Administrators
HA300	SAP HANA – Implementation & Modeling	Application Consultants Project team members Data Modeling Consultants
HA400	ABAP Programming for SAP HANA	Developers with experience in ABAP programming
HA450	Application Development for SAP HANA	Developers Development consultants
BW362	SAP BW on SAP HANA	Application consultants

Figure 127: SAP HANA Training System Access

We have fully-configured training systems in place to support all of these SAP HANA courses for your own personal access.

LESSON SUMMARY

You should now be able to:

- Develop your SAP HANA knowledge

Glossary

Active/Active-Read Enabled Mode
Use of a secondary, redundant server to handle read only, intensive queries thereby allowing the primary server to focus on read/write activities

AFL
Application Function Library is a repository of SAP supplied, ready made modules that can be used to build predictive analysis models and business applications

Analytic Privilege
Object to define the data access rules for users

Analytic View
Modeling object used to define a star schema based data model, but is now replaced by the calculation view.

Attribute
A column used to bring meaning to a measure, for example to aggregate by country or year and used to drill-down.

Attribute View
Modeling object used to define dimensions and their attributes that can be used stand-alone or integrated in a star schema, but is now replaced by the calculation view.

BICS
Business Intelligence Consumer Service is an SAP developed access technology used to tightly integrate SAP data sources such as SAP HANA with SAP analysis applications such as Design Studio for Excel.

Build
Generate the run-time objects from the design-time definitions for one of more objects at a time.

BWA
Business Warehouse Accelerator is an optional BW add-on that speeds up reporting from BW by using in-memory technology. It went on to become the origins of SAP HANA technology.

Cache
Data storage mechanism made up of hardware and software that holds frequently used data so that future requests can be server faster.

Calculation View
Core modeling object used to define data model of any type including dimension, cube, and cube star schema, and replaces attribute and analytic views.

CDS
Core Data Services is a scripting language used to define the persistence layer and also the data modeling layer with the ability to add rich semantics to provide technical and business meaning to data.

Cloud Foundry
Open source, industry standard cloud platform used to build, deploy, and run and scale applications now adopted by SAP HANA for XSA developments.

Container
Collector of all database run-time objects used by an application. It is a logical layer that sits above the physical database schema where the actual database objects reside, used only by XSA applications.

Cube
Multi-dimensional data set used in slice and dice analysis and is one of the types of calculation view.

Data Warehouse Foundation (DWF)
Toolset that can be installed in SAP HANA to provide developers with tools to build an SQL data warehouse completely on SAP HANA without the need for BW.

Database Explorer
A tool of Web IDE for SAP HANA that provides developers with direct access to the database of SAP HANA.

Delivery Unit
Used to transport development artifacts from one HANA system to another, used only in classic XS developments.

Dimension
Collection of attributes that belong together (for example, product color, weight, price) used to provide meaning to measures and can be shared across many calculation views.

Extended Storage
An optional disk based data store that is fully integrated into the SAP HANA database and often used to off-load less important data from memory (hot) to a warm store.

Flowgraph
Object used to define an SDI data flow using a graphical editor, and also used to define a predictive analysis model.

Function
Custom read-only SQL based script used to either return a tabular result set that can be used as a data source in a calculation view (table function), or to return a single value to be used as an input parameter (scalar function).

Gerrit
Web based code review tool that integrated with Git that supports a workflow approach to collaborative code review and approval and used with SAP HANA XSA development projects.

Git
Industry standard, and very popular source code version control solution used by SAP HANA XSA developers to easily share code and modeling artifacts, and replaces the internal source code versioning with SAP HANA XS..

Graph
A model that represent entities that are best described in a network, such as a social network (Facebook)

HaaS
HANA as a Service — access SAP HANA as a public cloud service

HDI
HANA Deployment Infrastructure — The deployment infrastructure used to build XSA applications, where an 'all or nothing' build approach applies

MDX
Multi Dimensional Expression language was developed by Microsoft to provide an alternative to SQL for queries that access multi-dimensional data sources to be written in a less complex way and with dedicated syntax.

Measure
A numerical value that represents money, quantities or general numbers that can be aggregated and used in calculations.

MTA
Multi Target Application is an SAP HANA single application that can be deployed both on premise and in the cloud without adjustment of the code.

Multistore table
A table that is partitioned across memory and extended (disk) layers.

Multitenancy
An architecture that supports the complete isolation of applications running on the same platform.

Native DataStore Object (NDSO)
A component of DWF an NDSO is a persistence object used to store data in an SQL driven data warehouse built on SAP HANA, and provides data load request logging and delta management.

OData
Open Data Protocol is a poplar and open standard introduced by Microsoft and supported by XS and XSA that allows a developer to expose a HANA table or view to an application by defining reusable query logic.

ODBC
Open Database Connectivity is a protocol originally developed by Microsoft to provide connectivity between applications to any relational database. It is a very popular method of connecting application to SAP HANA.

ODBO
OLE DB for OLAP was introduced by Microsoft to provide a connectivity interface between slice and dice analysis tools such as XL pivot tables, and multi-dimensional data models, such as cube calculation views

Organization
Used in XSA developments to group Spaces that enable sharing of resources among developers./

Package
Used to organize development artifacts in a hierarchical manner, used only in classic XS developments.

PAL
Predictive Analysis Library is a repository of SAP supplied, ready-to-use algorithms, and functions meant to be used in predictive analysis models.

Persistence layer

The layer built using database objects that physically store data as opposed to the virtual layer that sits above it.

Procedure
Customer-SQL-based script used to define reusable processing logic that can be used universally across HANA to read and also write to the database. Sometimes called a stored procedure.

Project
Collector of all design-time artifacts used in an application.

R
A popular open-source programming language used to develop statistical / predictive models that is integrated with SAP HANA.

Role
A collection of privileges that can be assigned to another role or a user

SAP BW/4HANA
SAP's next generation BW that is built to run only on SAP HANA.

SAP S/4HANA
SAP's next generation ERP suite that is built to run only on SAP HANA

SAP Vora
Big Data integrator based on Apache Spark used to add a processing layer on top of any Big Data store in order to combine Big Data with HANA data.

Scale out
Install SAP HANA across multiple parallel servers to share the processing and data storage, and maybe also to provide redundant standby servers to be used in case of primary server failure.

Scale up
Add more CPU and/or RAM to improve performance of SAP HANA

SDA
Smart Data Access — In-built SAP HANA technology to expose remote data source as virtual tables in the SAP HANA database.

SDI
Smart Data Integration — In-built component of SAP HANA used to integrate and harmonize data from single or multiple sources of any type, in batch or real time.

SDQ
Smart Data Quality — In-built component of SAP HANA used to improve data quality by enrichment and cleansing of data being loaded through SDI.

Series Data
Data that is collected at measurable intervals such as time

SHINE
SAP HANA Interactive Education is a prebuilt model application supplied and maintained by SAP to showcase what can be built in SAP HANA that developers can study and learn from.

SLT
SAP Landscape Transformation Server is SAP's popular NetWeaver replicator tool used to move data from one system to another (including SAP HANA) in real time, also used in SAP S/4HANA Central Finance solution.

SP
Service Pack relates to an individual component of SAP HANA, such as SAP HANA Live tools SP 1.0

Space
Used in XSA developments to provide different collections of shared resources to various projects.

SPS
Support Pack Stack is a two-digit number that identifies the release of SAP HANA, for example, SAP HANA SPS12.

SQL
Structured Query Language is used to read and write from relational databases. It is the world's most popular language for doing this.

SQL Script
The native database query language of SAP HANA that is an extension of standard SQL with additional support for column store, in-memory processing plus advanced data processing such as text, spatial

Studio
Interface used by developers and modelers and administrators for building XS data models and applications.

Synonym

An alias that is used to provide access to a database object that resides outside the local container of an XSA application, mostly used to access shared objects that should not be part of an application ownership.

TensorFlow
Machine learning models provided by Google that can be accessed by SAP HANA

Web IDE for SAP HANA
Web Interface used by developers and modelers for building XSA data models and applications.

Web-based Development Workbench
Web interface used by developers and modelers to build XS data models and applications, and a lightweight alternative to Studio.

Workspace
A design time area used by a developer to build applications and models.

XS
Extended Application Services is SAP's first generation framework that provides all components needed to build and run applications completely in SAP HANA.

XSA
Extended Application Services — Advanced is SAP's second generation framework that provides all components needed to build and run applications completely in SAP HANA and now follows common cloud standards for more flexible deployment options.